TEORÍA DE DECISIÓN

Estas transparencias contienen material adaptado del curso de PATTERN RECOGNITION AND MACHINE LEARNING de Heikki Huttunen
Statistical Signal Processing Detection Theory - Steven M. Kay

TEORÍA DE DECISIÓN

- Tiene muchos temas comunes con aprendizaje automático
- Los métodos se basan en teoría de estimación y tratan dar respuesta a cuestiones como:

¿Está unas señal específica presente en nuestra serie temporal ?
Ej: detección de un tono en presencia de ruido

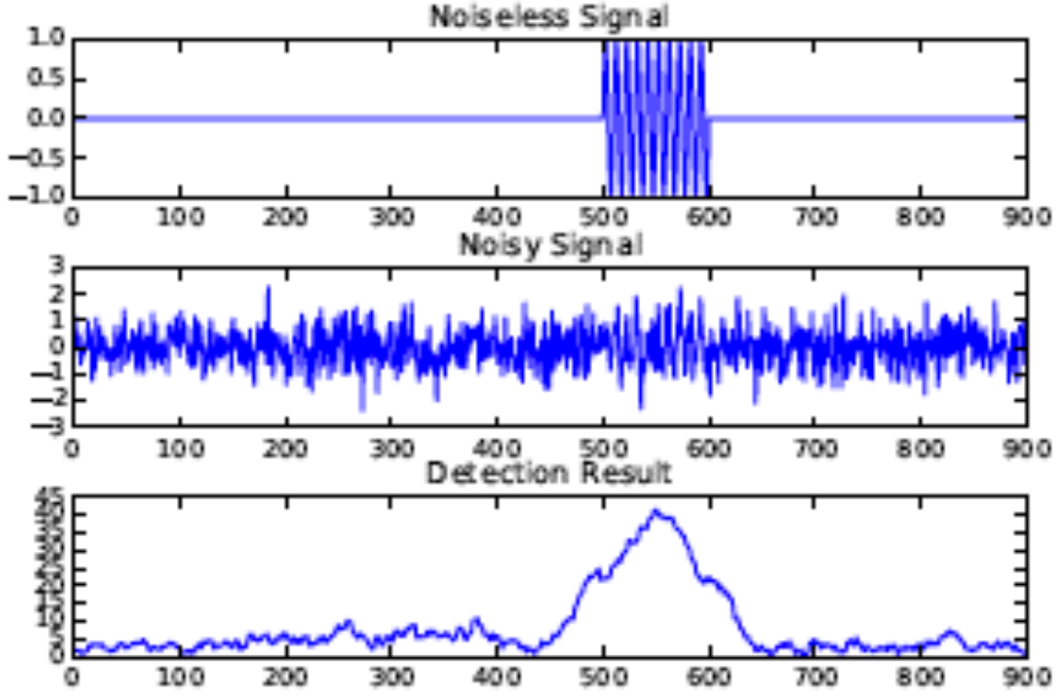
¿ Cambia el valor de continua de una señal en el tiempo?

¿Está presente una persona en este cuadro de video?

- Relacionado con test de hipótesis que es muy aplicado en medicina: ¿La respuesta se debe a la nueva medicación o a variaciones aleatorias?

EJ: DETECCIÓN DE SINUSOIDE EN PRESENCIA DE RUIDO

$$\mathcal{H}_1 : x[n] = A\cos(2\pi f_0 n + \phi) + w[n]$$
$$\mathcal{H}_0 : x[n] = w[n]$$



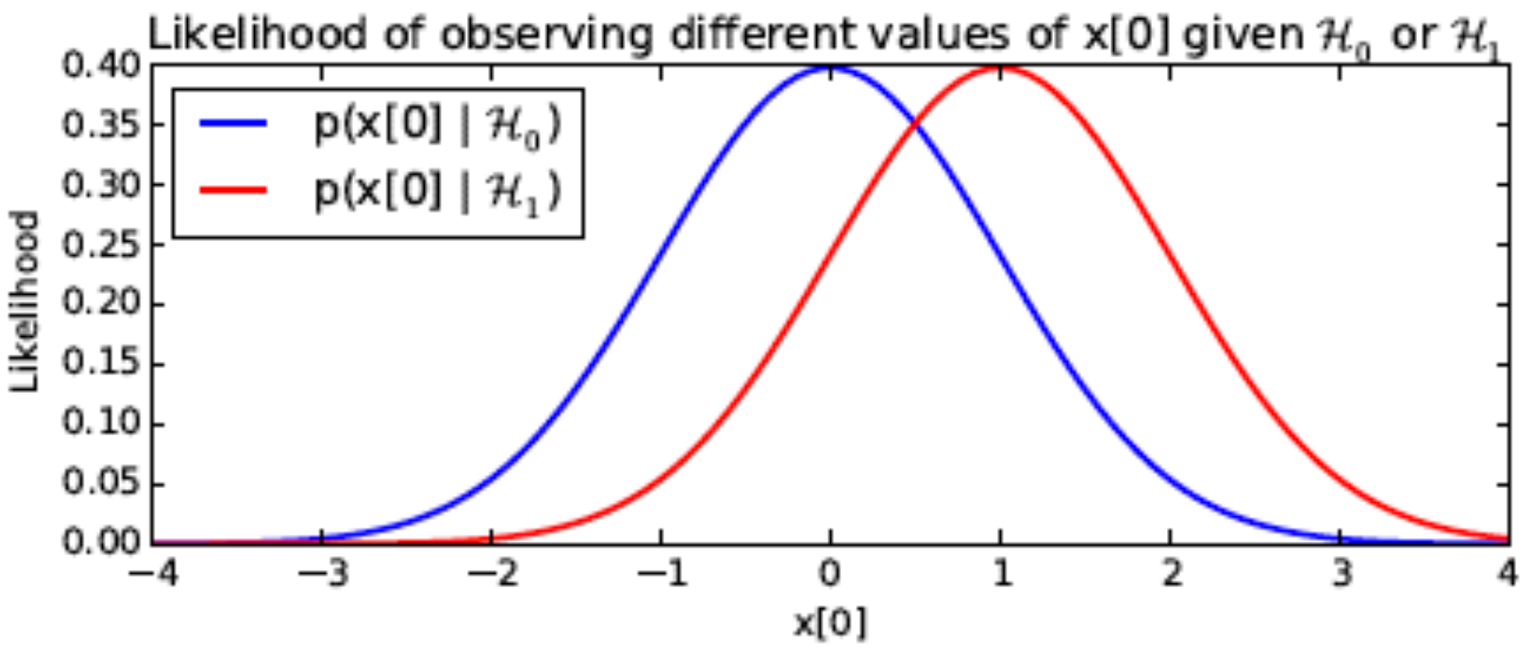
- H1: corresponde a la hipótesis señal presente - hipótesis alternativa
- H0: corresponde a la hipótesis solo ruido- hipótesis nula

EJEMPLO : DETECCIÓN BINARIA EN PRESENCIA DE RUIDO

- Supongamos que x[0] es una muestra que proviene de una de dos densidades *N(0,1)* o *N(1,1)*
- El objetivo es elegir la hipótesis correcta en forma óptima.
- Las hipótesis son en este caso:

$$\mathcal{H}_1 : \mu = 1,$$
$$\mathcal{H}_0 : \mu = 0,$$

elijo la hipótesis con mayor verosimilitud para x[0] observado.



EJEMPLO : DETECCIÓN BINARIA EN PRESENCIA DE RUIDO

$$\mathcal{H}_1 : p(x[0] \mid \mu = 1) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x[n]-1)^2}{2}\right)$$

$$\mathcal{H}_0 : p(x[0] \mid \mu = 0) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x[n])^2}{2}\right).$$

- Elijo H_1 si $"\mu = 1"$es más probable que $"\mu = 0"$ es decir si:

$$p(x[0] \mid \mu = 1) > p(x[0] \mid \mu = 0)$$

EJEMPLO : DETECCIÓN BINARIA EN PRESENCIA DE RUIDO

- La regla de decisión se puede reducir a comparar el valor de x[0] con el umbral = 1/2.
- Considerando que:

$$p(x[0] \mid \mu = 1) > p(x[0] \mid \mu = 0) \iff \frac{p(x[0] \mid \mu = 1)}{p(x[0] \mid \mu = 0)} > 1$$

- La regla de decisión consiste en comparar el valor del cociente de verosimilitudes (test de razón de verosimilitud, LRT) contra un umbral, en este caso de valor 1.

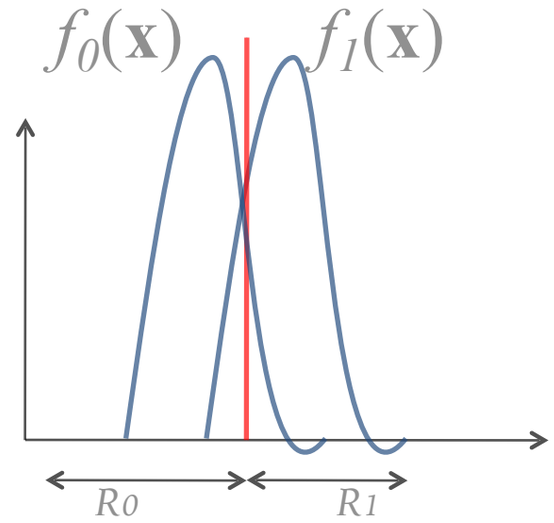
EJEMPLO: DETECCIÓN DE LESIÓN

Consideremos test de hipótesis simple :

$H_0 : \mathbf{x} \approx f_0(\mathbf{x})$ hipótesis nula, "lesión benigna", normal

$H_1 : \mathbf{x} \approx f_1(\mathbf{x})$ evento a detectar, "lesión maligna", enfermo

R_i : región donde se decide H_i $i = 0,1$



P_{FA} = probabilidad de falsa alarma $= \int_{R_1} f_0(\mathbf{x}) d\mathbf{x}$

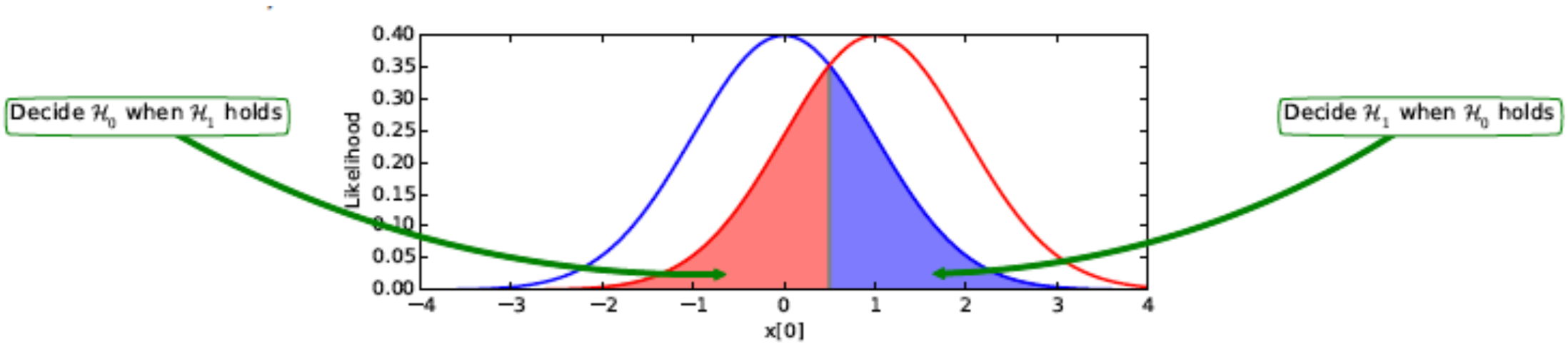
P_D = probabilidad de detección $= \int_{R_1} f_1(\mathbf{x}) d\mathbf{x}$

Al ser las densidades positivas si aumenta P_D aumenta P_{FA}

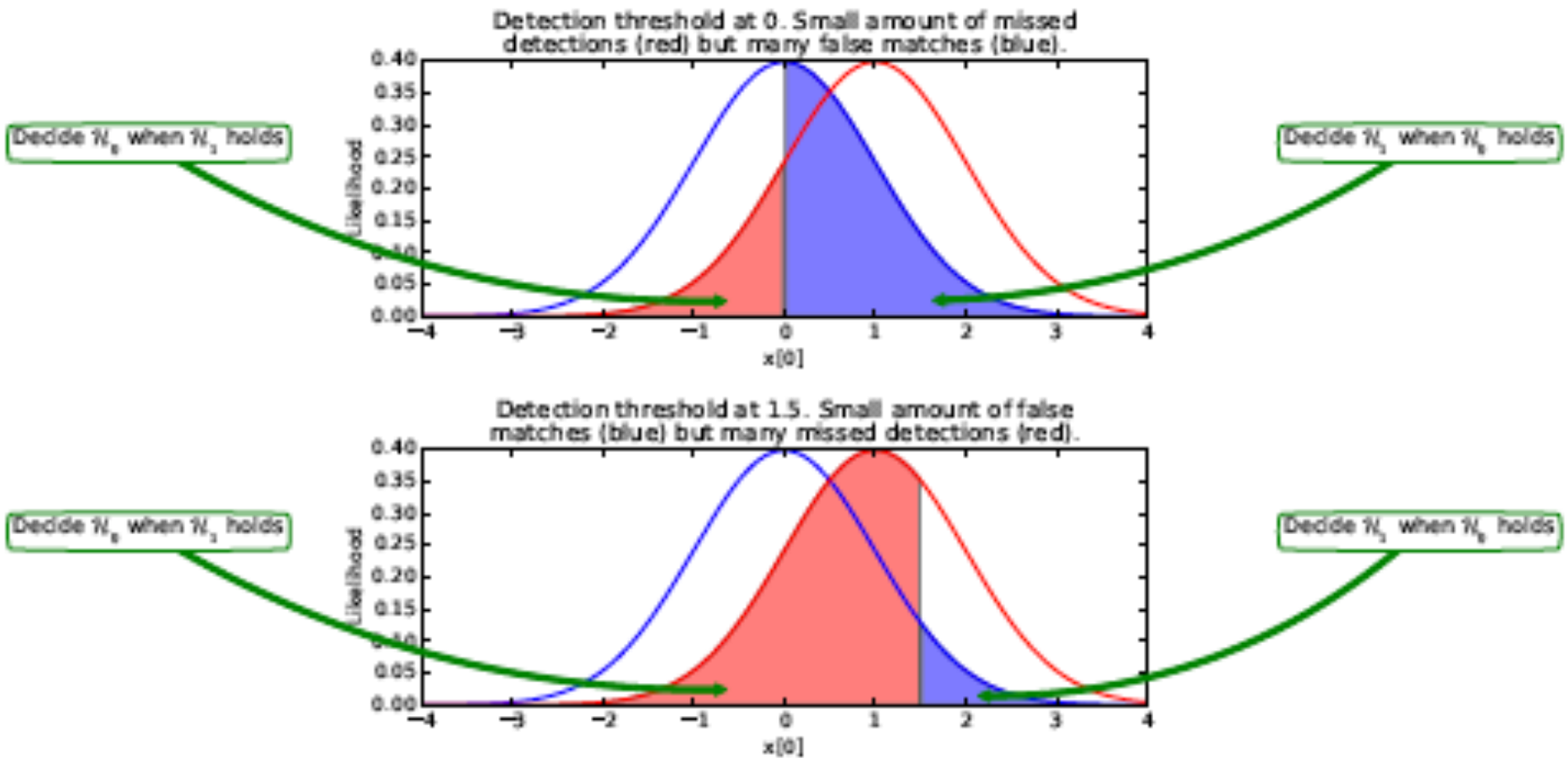
Solo puede ser $P_D = 1$ y $P_{FA} = 0$ si las densidades son disjuntas

TIPOS DE ERROR

- En el ejemplo anterior se ve que aparecen distintos tipos de error y con costos distintos, perder una detección es mucho más caro que una falsa detección.
- El compromiso entre falsas detecciones (área azul) y pérdidas (área roja) se puede ajustar cambiando el umbral de detección.



TIPOS DE ERROR



- Cada probabilidad se puede hacer arbitrariamente pequeña ajustando el umbral.

EJEMPLO: PROBABILIDAD FALSA ALARMA, PÉRDIDA , DETECCIÓN

- Probabilidad de falsa alarma y perdida cuando $\gamma = 1.5$

$$P_{FA} = P(x[0] > \gamma \mid \mu = 0) = \int_{1.5}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x[n])^2}{2}\right) dx[n] \approx 0.0668.$$

$$P_M = P(x[0] < \gamma \mid \mu = 1) = \int_{-\infty}^{1.5} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x[n]-1)^2}{2}\right) dx[n] \approx 0.6915.$$

- Probabilidad de detección:

$$P_D = 1 - P_M = \int_{1.5}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x[n]-1)^2}{2}\right) dx[n] \approx 0.3085.$$

TEOREMA DE NEYMAN-PEARSON

- Teniendo en cuenta que P_{FA} y P_D dependen cada una de la otra nos interesa maximizar P_D sujeto a un valor máximo permitido de P_{FA}.
- Teorema de Neyman-Pearson: establece que fijada P_{FA} se maximiza P_D si :

$$L(\mathbf{x}) = \frac{p(\mathbf{x}; \mathcal{H}_1)}{p(\mathbf{x}; \mathcal{H}_0)} > \gamma_L$$

donde el umbral cumple:

$$\int_{\mathbf{x}:L(\mathbf{x})>\gamma_L} p(\mathbf{x}; \mathcal{H}_0)\, d\mathbf{x} = P_{FA}.$$

EJEMPLO: NEYMAN-PEARSON DETECCIÓN BINARIA

- Quiero encontrar el mejor detector cuando trasmito señal con media 1 o 0 en presencia de ruido gaussiano y quiero que el % de falsas alarmas no supere el 10%. ($P_{FA}=0.1$)
- Aplicando Neyman-Pearson la regla de detección es:

$$\text{Select } \mathcal{H}_1 \text{ if } \frac{p(x \mid \mu = 1)}{p(x \mid \mu = 0)} > \gamma_L$$

- Para resolver el problema tenemos que determinar el umbral:

$$\int_{\gamma}^{\infty} p(x \mid \mu = 0)\, dx = 0.1. \qquad \gamma_L = L(\gamma)$$

EJEMPLO: NEYMAN-PEARSON DETECCIÓN BINARIA

- Esto puede implementarse con la función isf, que resuelve la inversa de la función distribución acumulada:
- Los parámetros loc y scale corresponden a la media y la desviación estándar de la densidad gaussiana.

```
>>> import scipy.stats as stats

>>> # Compute threshold such that P_FA = 0.1
>>> T = stats.norm.isf(0.1, loc = 0, scale = 1)
>>> print T
1.28155156554
```

DETECTOR NP PARA UNA SEÑAL CONOCIDA

- Neyman-Pearson aplica para cualquier problema en el que tenemos acceso a las verosimilitudes.
- Un ejemplo es el caso en que se trasmite una señal conocida y se detecta la reflejada con ruido superpuesto un tiempo después(radar).

$$\mathcal{H}_1 : x[n] = s[n] + w[n] \qquad n = 0, 1, \ldots, N-1$$
$$\mathcal{H}_0 : x[n] = w[n]. \qquad n = 0, 1, \ldots, N-1$$

DETECTOR NP PARA UNA SEÑAL CONOCIDA

- En este caso las verosimilitudes, bajo el supuesto de ruido gaussiano con media nula corresponden con pdf de gaussianas multivariadas:

$$p(\mathbf{x} \mid \mathcal{H}_1) = \prod_{n=0}^{N-1} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x[n]-s[n])^2}{2\sigma^2}\right),$$

$$p(\mathbf{x} \mid \mathcal{H}_0) = \prod_{n=0}^{N-1} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x[n])^2}{2\sigma^2}\right).$$

- El test de cociente de verosimilitudes:

$$\frac{p(\mathbf{x} \mid \mathcal{H}_1)}{p(\mathbf{x} \mid \mathcal{H}_0)} = \exp\left[-\frac{1}{2\sigma^2}\left(\sum_{n=0}^{N-1}(x[n]-s[n])^2 - \sum_{n=0}^{N-1}(x[n])^2\right)\right] > \gamma_L$$

DETECTOR NP PARA UNA SEÑAL CONOCIDA

- Tomando logaritmo de ambos lados y juntando la señal conocida con el umbral se simplifica la regla de detección:

$$-\frac{1}{2\sigma^2}\left(\sum_{n=0}^{N-1}(x[n]-s[n])^2-\sum_{n=0}^{N-1}(x[n])^2\right)>\ln\gamma_L$$

$$\frac{1}{\sigma^2}\sum_{n=0}^{N-1}x[n]s[n]-\frac{1}{2\sigma^2}\sum_{n=0}^{N-1}(s[n])^2>\ln\gamma_L.$$

$$\sum_{n=0}^{N-1}x[n]s[n]>\sigma^2\ln\gamma_L+\frac{1}{2}\sum_{n=0}^{N-1}(s[n])^2.$$

$$\sum_{n=0}^{N-1}x[n]s[n]>\gamma'.$$

NEYMAN PEARSON DETECTOR DE CONTINUA

$$\frac{1}{N}\sum_{n=0}^{N-1} x[n] > \frac{\sigma^2}{NA}\ln\gamma_L + \frac{A}{2} = \gamma'.$$

$$\begin{aligned} P_{FA} &= \Pr\{T(\mathbf{x}) > \gamma'; \mathcal{H}_0\} \\ &= Q\left(\frac{\gamma'}{\sqrt{\sigma^2/N}}\right) \end{aligned}$$

- Regla de detección: Comparo la media de las muestras recibidas contra un umbral que depende del PFA elegido.
- T(x) tiene distribución Gaussiana bajo ambas hipótesis con media 0 y A y varianza N veces menor.

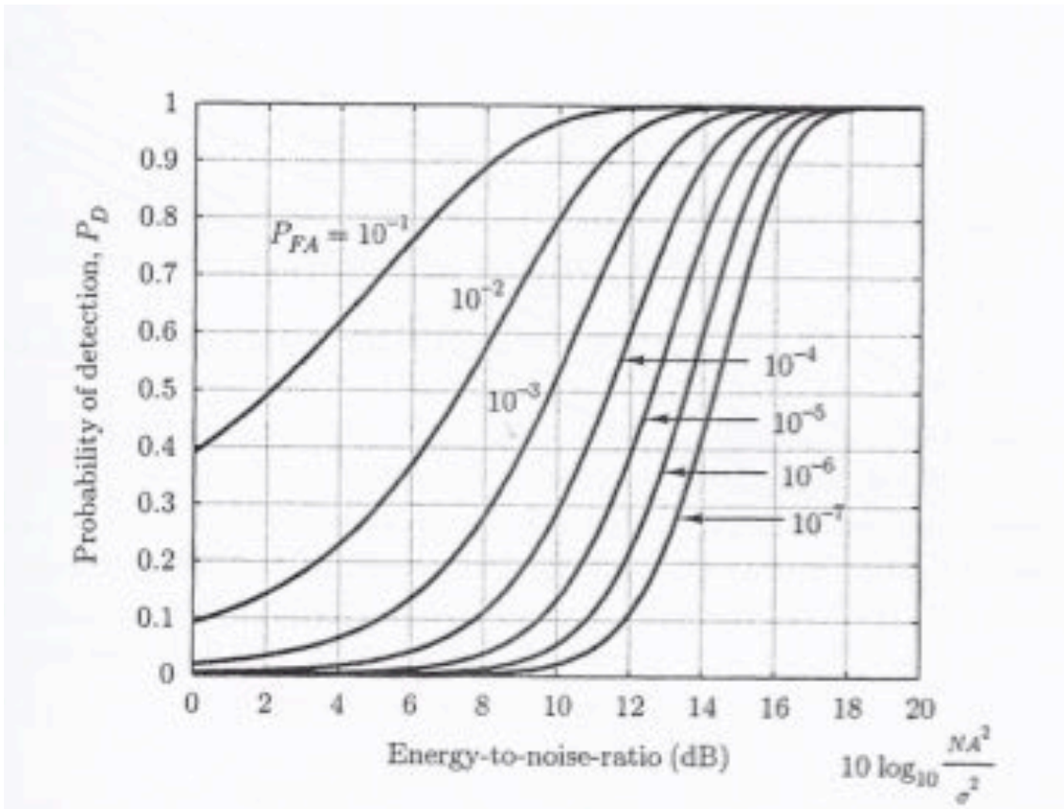
Ver Kay

NEYMAN PEARSON DETECTOR DE CONTINUA

$$
\begin{aligned}
P_D &= \Pr\{T(\mathbf{x}) > \gamma'; \mathcal{H}_1\} \\
&= Q\left(\frac{\gamma' - A}{\sqrt{\sigma^2/N}}\right).
\end{aligned}
$$

$$
\gamma' = \sqrt{\frac{\sigma^2}{N}} Q^{-1}(P_{FA})
$$

$$
\begin{aligned}
P_D &= Q\left(\frac{\sqrt{\sigma^2/N}Q^{-1}(P_{FA}) - A}{\sqrt{\sigma^2/N}}\right) \\
&= Q\left(Q^{-1}(P_{FA}) - \sqrt{\frac{NA^2}{\sigma^2}}\right).
\end{aligned}
$$



Kay

NEYMAN PEARSON (DISTINTAS MEDIAS – IGUAL VARIANZA)

$$
\begin{aligned}
P_{FA} &= \Pr\{T > \gamma'; \mathcal{H}_0\} \\
&= Q\left(\frac{\gamma' - \mu_0}{\sigma}\right) \\
P_D &= \Pr\{T > \gamma'; \mathcal{H}_1\} \\
&= Q\left(\frac{\gamma' - \mu_1}{\sigma}\right) \\
&= Q\left(\frac{\mu_0 + \sigma Q^{-1}(P_{FA}) - \mu_1}{\sigma}\right) \\
&= Q\left(Q^{-1}(P_{FA}) - \left(\frac{\mu_1 - \mu_0}{\sigma}\right)\right)
\end{aligned}
$$

have

$$P_D = Q\left(Q^{-1}(P_{FA}) - \sqrt{d^2}\right)$$

Kay

NEYMAN PEARSON (VARIANZAS DISTINTAS)

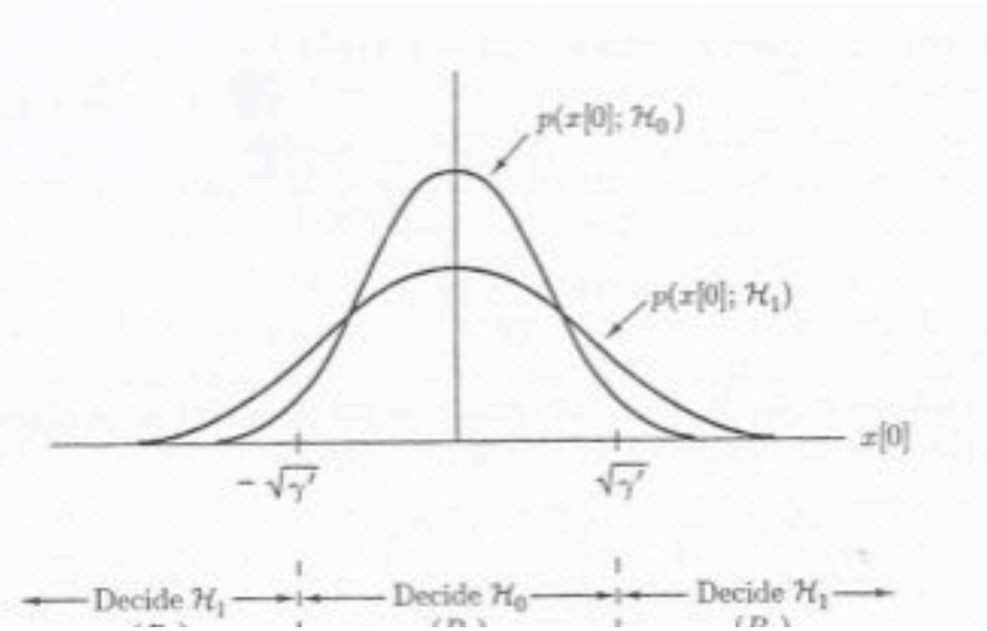
$$-\frac{1}{2}\left(\frac{1}{\sigma_1^2}-\frac{1}{\sigma_0^2}\right)\sum_{n=0}^{N-1}x^2[n] > \ln\gamma + \frac{N}{2}\ln\frac{\sigma_1^2}{\sigma_0^2}.$$

ve have

$$\frac{1}{N}\sum_{n=0}^{N-1}x^2[n] > \gamma'$$

$$\gamma' = \frac{\frac{2}{N}\ln\gamma + \ln\frac{\sigma_1^2}{\sigma_0^2}}{\frac{1}{\sigma_0^2}-\frac{1}{\sigma_1^2}}.$$

Kay



DETECTOR DE UNA SINUSOIDE CONOCIDA EN PRESENCIA DE RUIDO

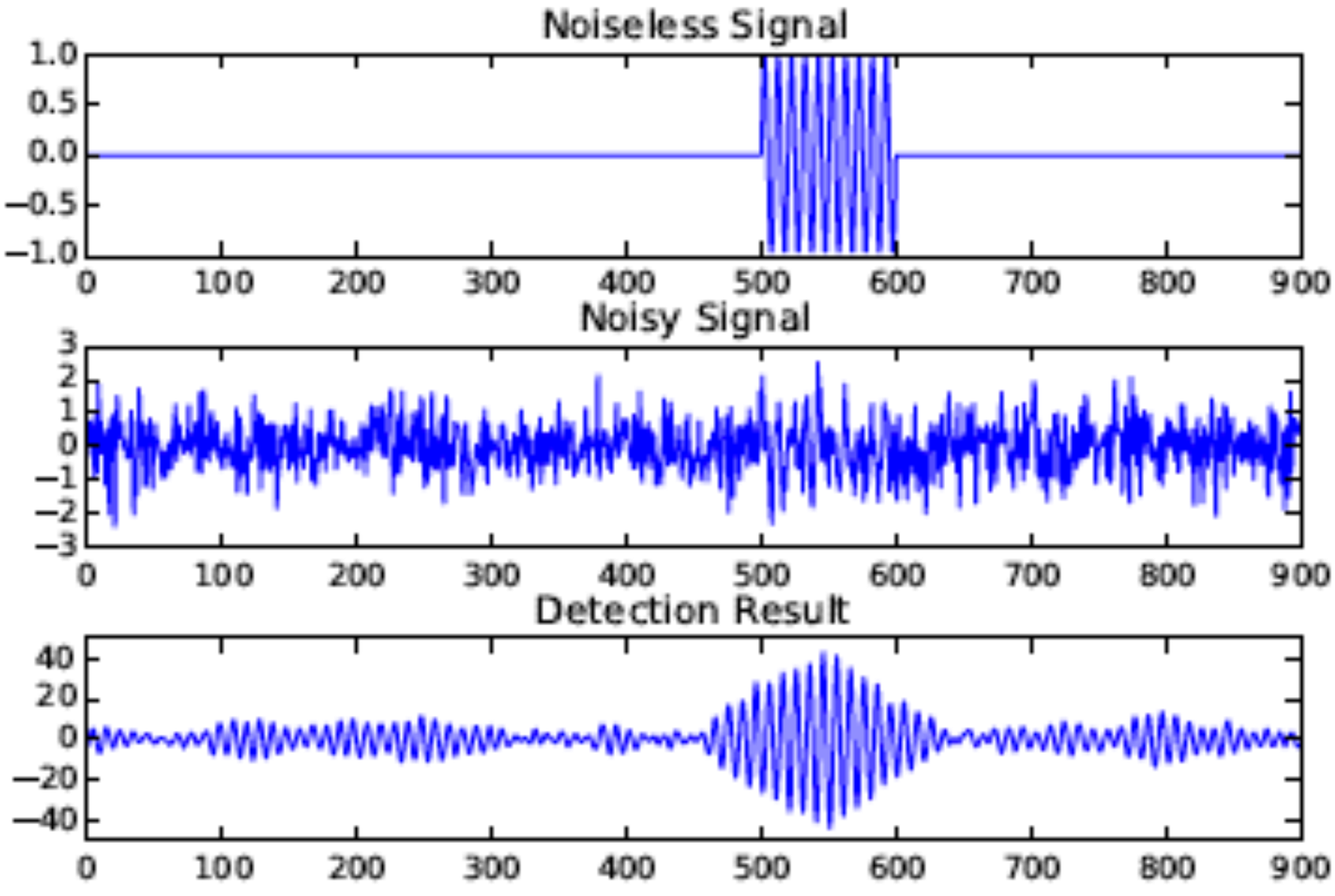
$$\sum_{n=0}^{N-1} x[n] A\cos(2\pi f_0 n + \phi) > \gamma \Rightarrow A\sum_{n=0}^{N-1} x[n]\cos(2\pi f_0 n + \phi) > \gamma.$$

- Regla de detección: comparo la correlación de la señal recibida con la señal conocida y comparo con un umbral que depende de del P_{FA} elegido.

$$\sum_{n=0}^{N-1} x[n]\cos(2\pi f_0 n + \phi) > \gamma'.$$

DETECTOR DE UNA SINUSOIDE CONOCIDA EN PRESENCIA DE RUIDO

- Ejemplo con desviación 0.5.



DETECCIÓN DE UNA SEÑAL ALEATORIA EN PRESENCIA DE RUIDO

- En el caso previo se suponía que conocemos tanto la frecuencia como la fase de la sinusoide algo que agrega muchas restricciones al problema de detección.
- Es más razonable suponer que no conocemos exactamente la señal pero si la estructura de correlación $\mathbf{s} \sim \mathcal{N}(0, \mathbf{C}_s)$

y formular el problema de detección como el test de hipótesis:

$$\mathcal{H}_0 : \mathbf{x} \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$
$$\mathcal{H}_1 : \mathbf{x} \sim \mathcal{N}(0, \mathbf{C}_s + \sigma^2 \mathbf{I})$$

DETECCIÓN DE UNA SEÑAL ALEATORIA EN PRESENCIA DE RUIDO

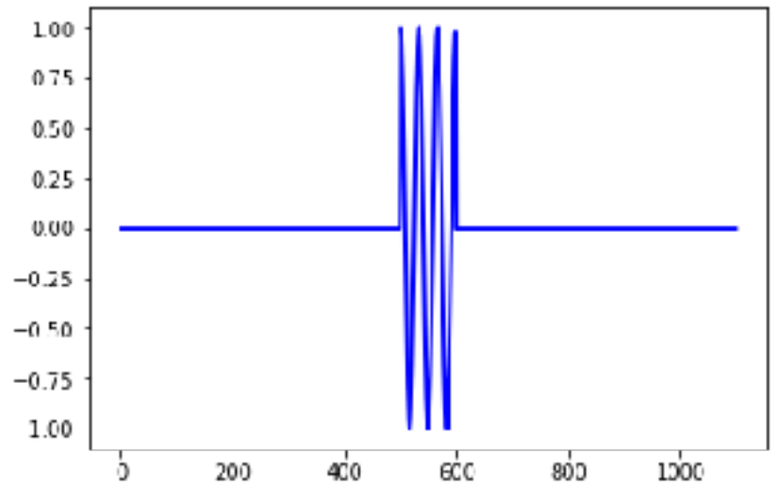
- La regla de decisión: $\text{Decide } \mathcal{H}_1, \text{ if } \mathbf{x}^T\hat{\mathbf{s}} > \gamma,$

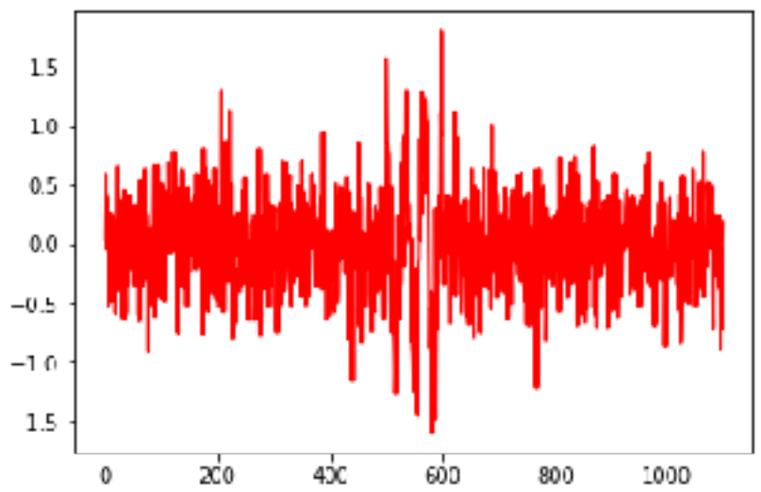
Con: $$\hat{\mathbf{s}} = \mathbf{C}_s(\mathbf{C}_s + \sigma^2\mathbf{I})^{-1}\mathbf{x}.$$

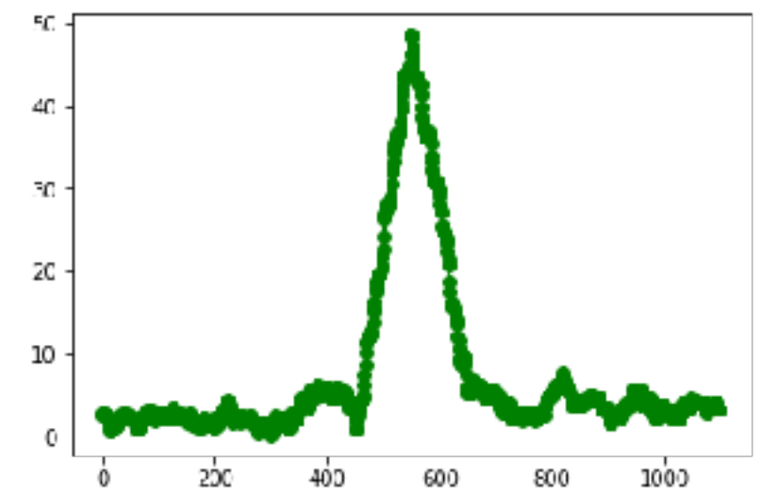
Para el caso de la sinusoide: $$\left|\sum_{n=0}^{N-1} x[n]\exp(-2\pi i f_0 n)\right| > \gamma.$$

```
import numpy as np

h = np.exp(-2 * np.pi * 1j * f0 * n)
y = np.abs(np.convolve(h, xn, 'same'))
```

DETECCIÓN SINUSOIDE CON FASE ALEATORIA EN RUIDO

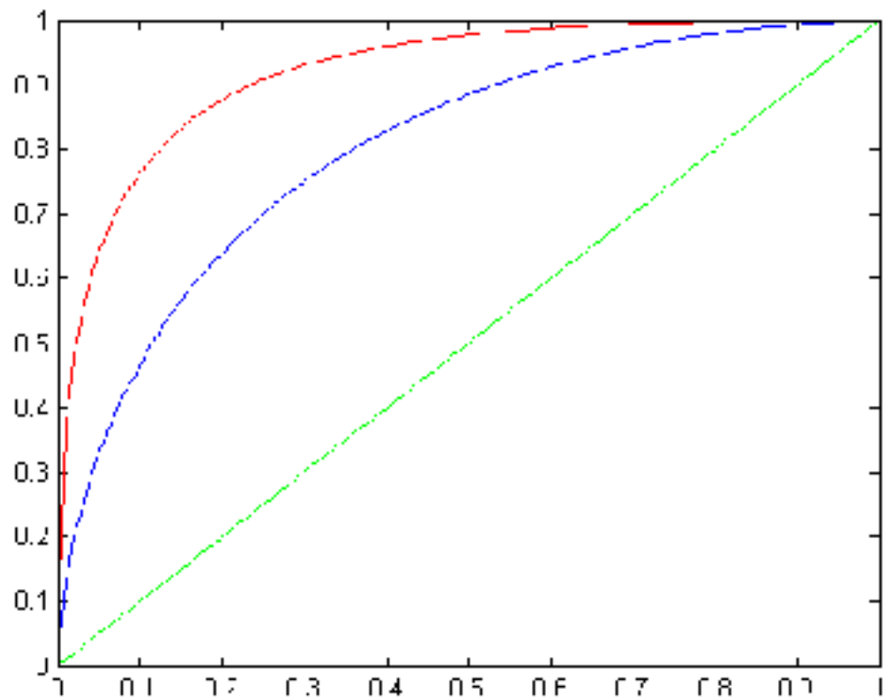






RECEIVER OPERATING CHARACTERISTIC (ROC CURVE)

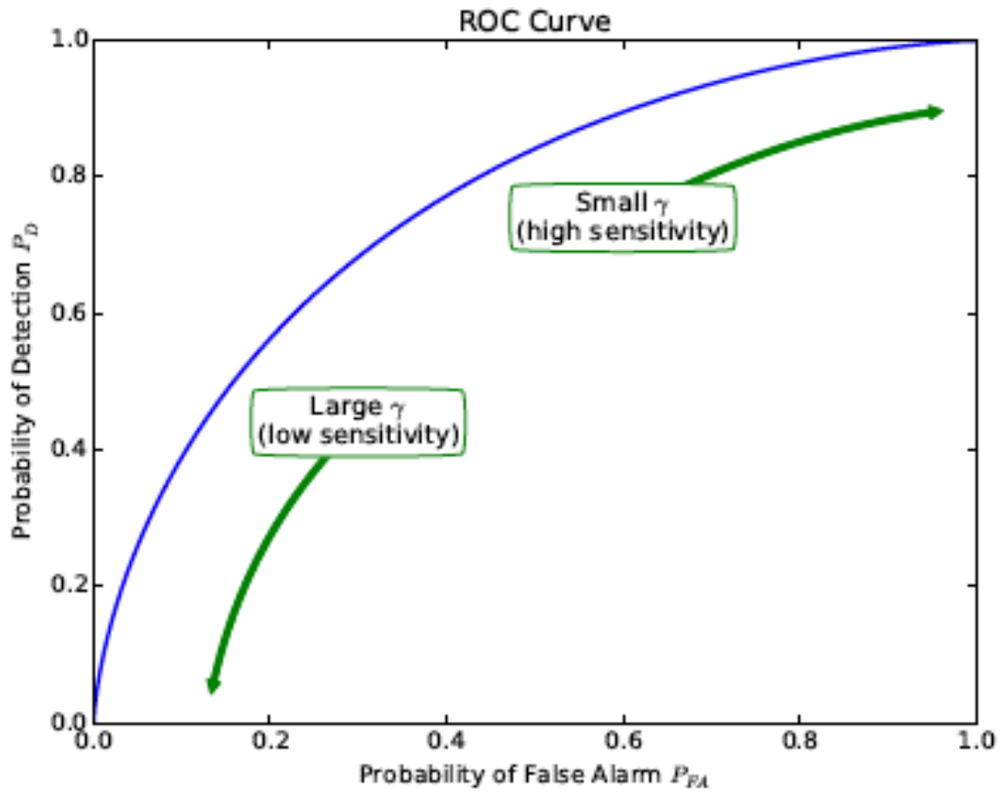
- Una forma usual de mostrar el desempeño de un detector es a través de la curva característica de operación (ROC)
- Describe la relación entre P_{FA} y P_D para todos los valores del umbral.
- La forma de la ROC depende del problema y del detector seleccionado.



EJEMPLO: ROC DETECCIÓN NIVEL DE DC

$$P_D(\gamma) = \int_{\gamma}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x-1)^2}{2}\right) dx$$

$$P_{FA}(\gamma) = \int_{\gamma}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) dx$$



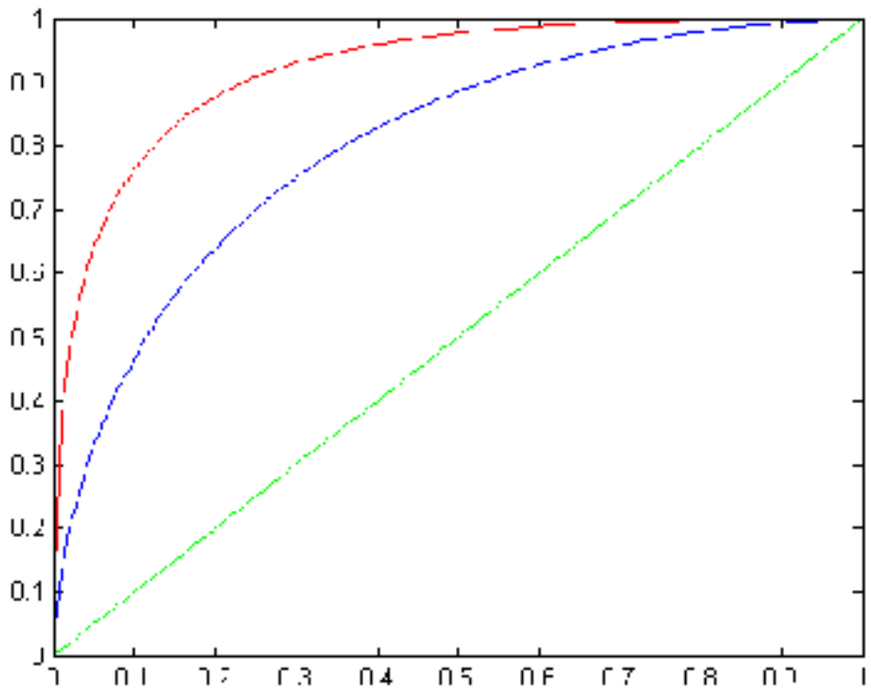
$$P_D(\gamma) = \int_{\gamma-1}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) dx = P_{FA}(\gamma - 1).$$

ROC

Ej: $f_0 = N(\mu_0, \sigma^2)$ $f_1 = N(\mu_1, \sigma^2)$

ROC determinada por la "discriminabilidad"

$$d = \frac{|\mu_1 - \mu_0|}{\sigma}$$



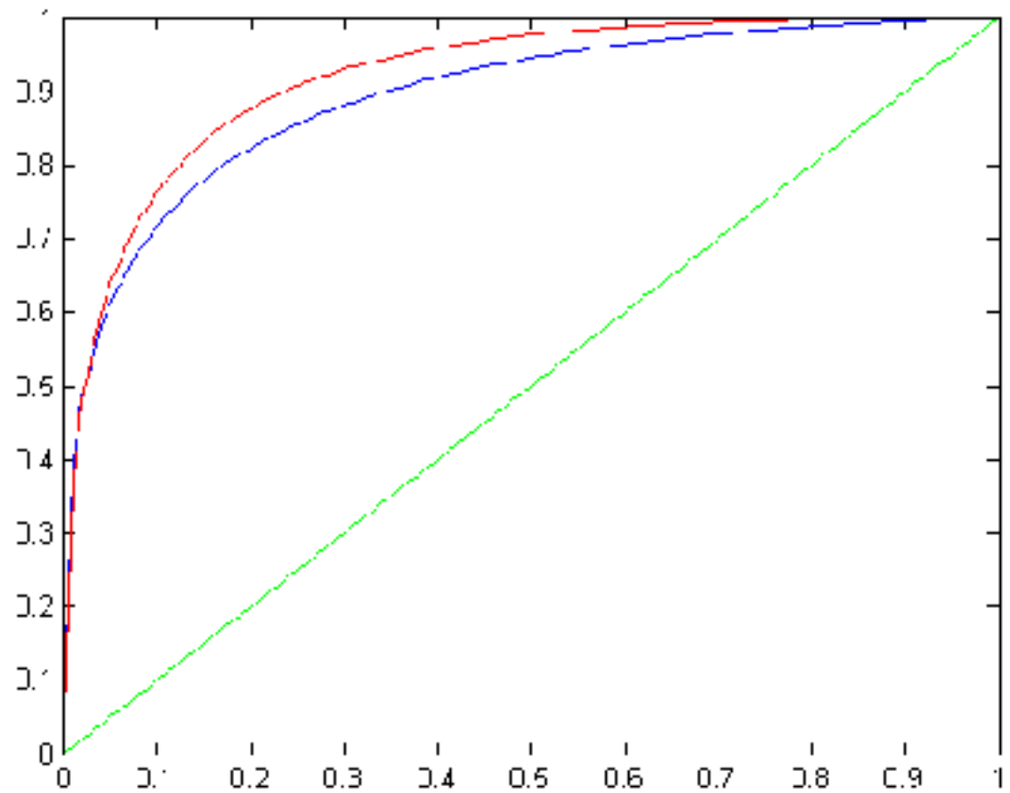
$\mu_{01} = 0$ $\mu_{11} = 1$ $\sigma_{01} = \sigma_{11} = 1$ roja

$\mu_{02} = 0$ $\mu_{12} = .8$ $\sigma_{02} = 1$ $\sigma_{12} = .8$ azul

$\mu_{01} = 0$ $\mu_{11} = 1$ $\sigma = 1$ roja

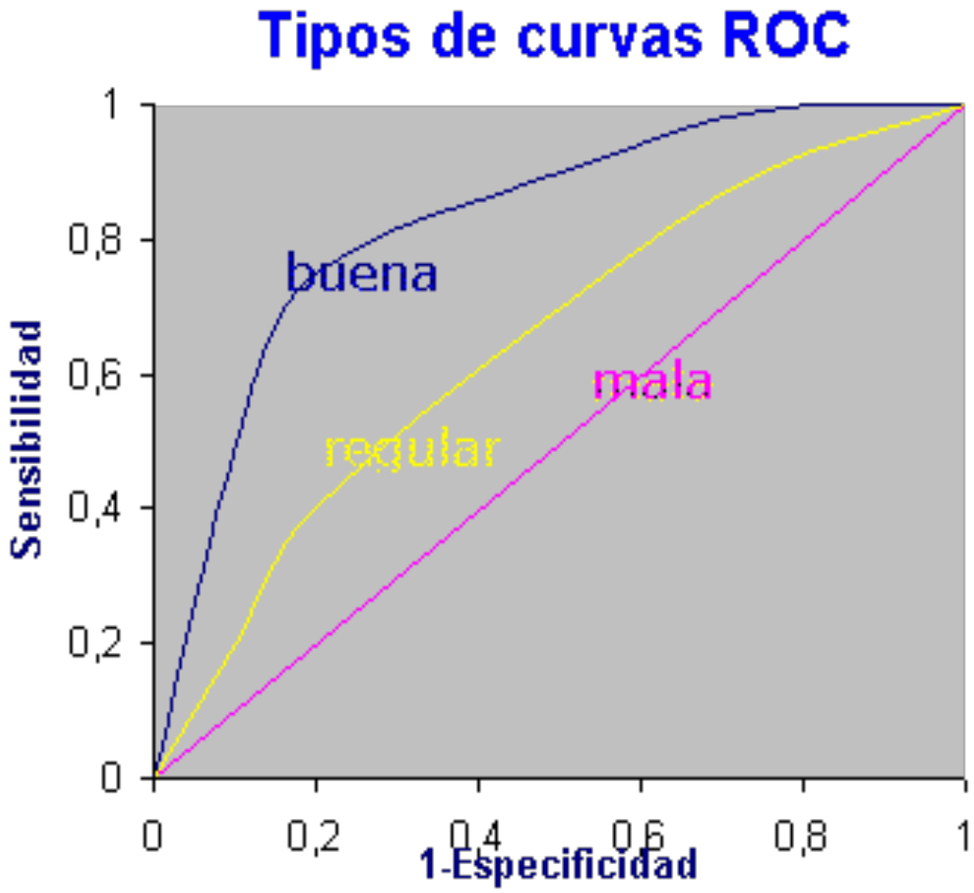
$\mu_{02} = 0$ $\mu_{12} = .6$ $\sigma = 1$ azul

$d_1 = 1$ $d_2 = .4$

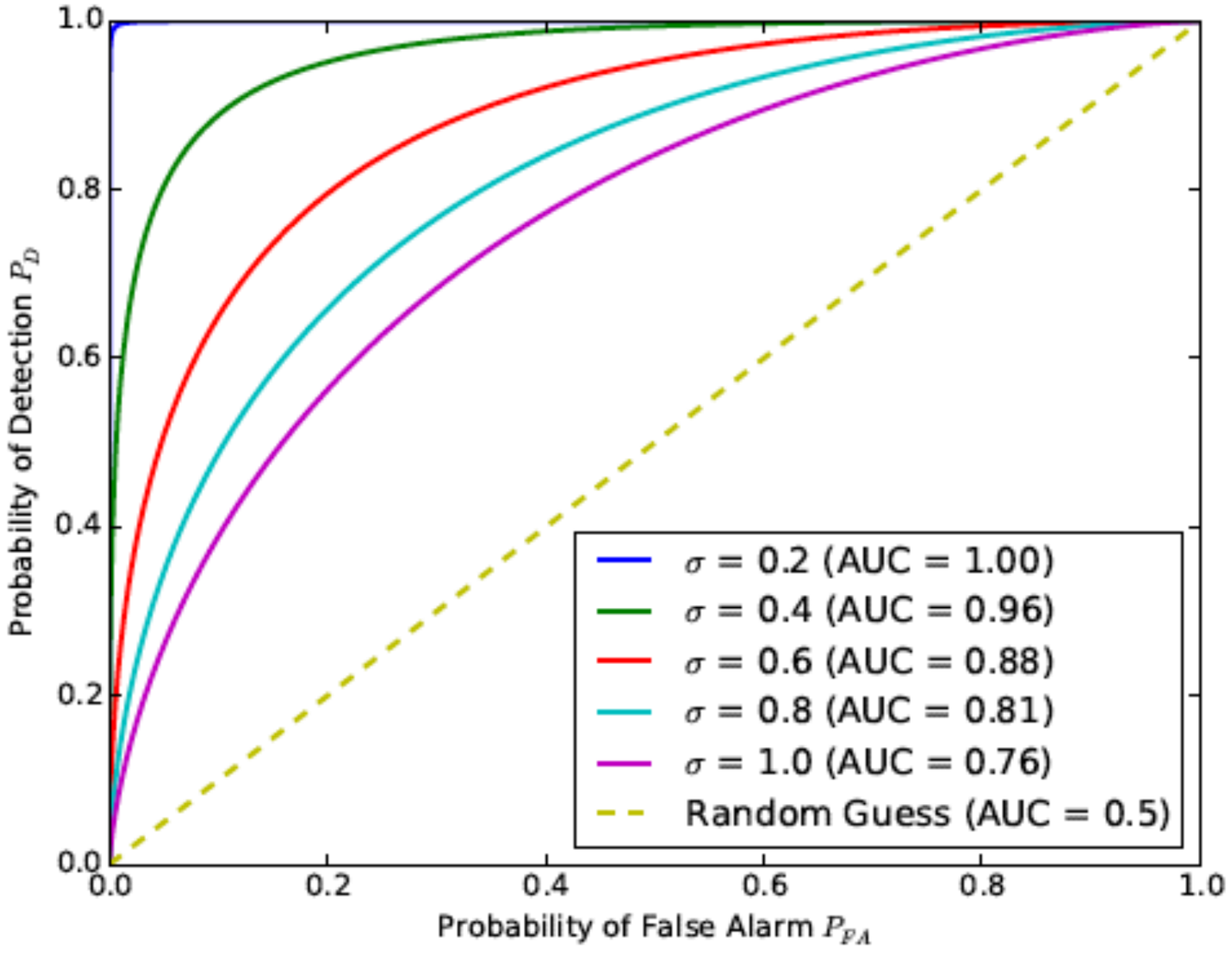


ROC Y AUC

- Cuanto mayores son los valores de la ROC mejor es el desempeño del detector.
- La diagonal corresponde a suposición aleatoria
- Una medida de desempeño que se deriva de la ROC es el Área debajo de la curva ROC que se denomina AUC
- Es una medida que es independiente del umbral y que representa el desempeño del clasificador para todos los umbrales.
- En el ejemplo de la detección de continua el AUC disminuye cuando aumenta la varianza del ruido.



ROC Y AUC


1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Probability of Detection P_D
Probability of False Alarm P_{FA}
σ = 0.2 (AUC = 1.00)
σ = 0.4 (AUC = 0.96)
σ = 0.6 (AUC = 0.88)
σ = 0.8 (AUC = 0.81)
σ = 1.0 (AUC = 0.76)
Random Guess (AUC = 0.5)

AUC EMPÍRICA

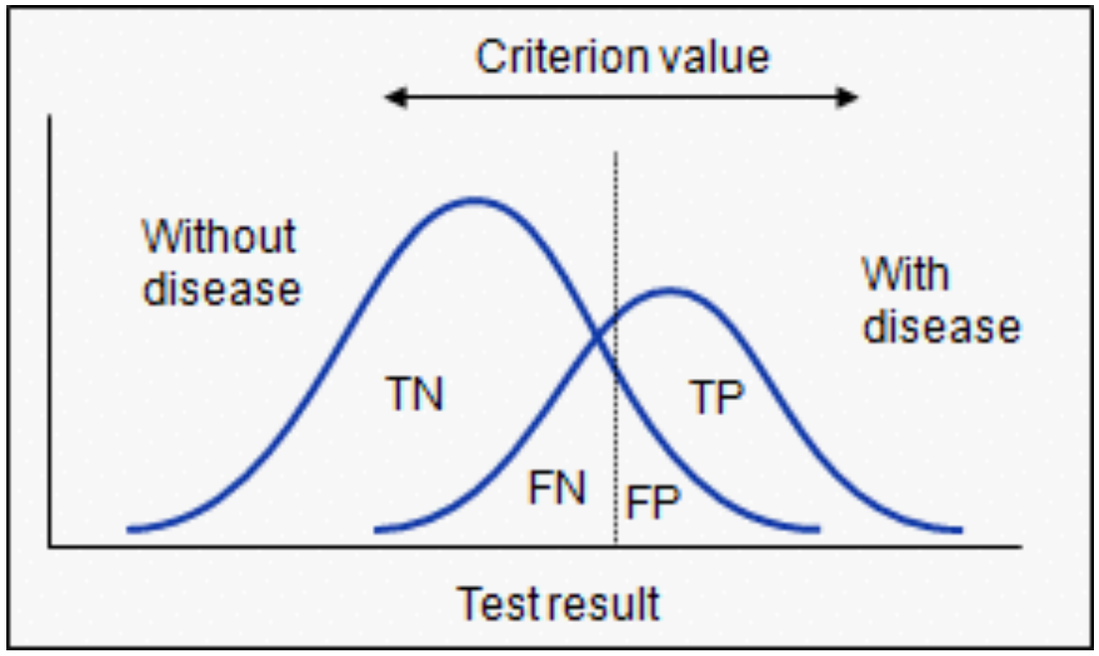
- El uso de la ROC y AUC surge en problemas de detección de objetivos con radar o radio.
- Más recientemente se han convertido en medidas usuales del desempeño de clasificadores, y en particular son medidas muy usadas en el área médica para comparar la efectividad de distintos test o tratamientos. Dependiendo del área de aplicación pueden variar la terminología utilizada.
- Usualmente no se tiene la posibilidad de usar formas cerradas analíticas para determinar la ROC y AUC, siendo lo usual evaluar los resultados predichos en un conjunto de test (TP, FP, TN, FN, con Matriz de Confusión)

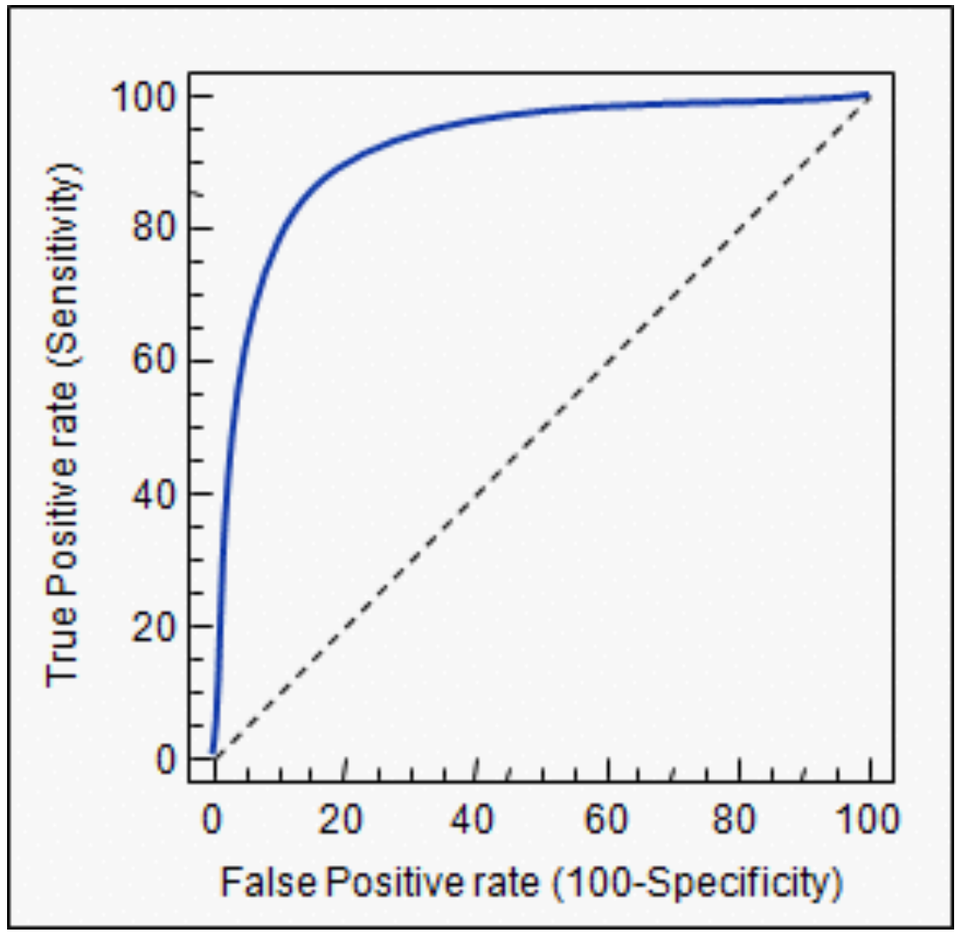
TÉRMINOS ESTADÍSTICOS PARA TEST DE HIPÓTESIS

Estadísticos	Ingenieros
Test estadístico y umbral	Detector
Hipótesis nula H_0	Hipótesis solo ruido
Hipótesis alternativa H_1	Hipótesis señal + ruido
Región crítica	Región con presencia de señal
Error tipo I (decido H_1 cuando H_0	Falsa alarma
Error tipo II (decido H_0 cuando H_1	Pérdida
Nivel de significancia o tamaño del	Probabilidad de Falsa alarma (P_{FA})
Probabilidad de Error tipo II (β)	Probabilidad de pérdida(P_M)
Potencia del test (1-β)	Probabilidad de detección(P_D)

APLICACIÓN ROC A DIAGNÓSTICO MÉDICO

Ej: comparación de dos test de diagnóstico de diferentes laboratorios





Sensibilidad: *probabilidad de que un test resulte positivo cuando la enfermedad esta presente.*

Especificidad: *probabilidad que un test resulte negativo cuando la enfermedad no está presente*

DETERMINACIÓN EMPÍRICA P_D Y P_{FA}

➢ Analizo en forma experimental utilizando un conjunto de test.

	Predicción +	Predicción -
Clase +	TP	FN
Clase -	FP	TN

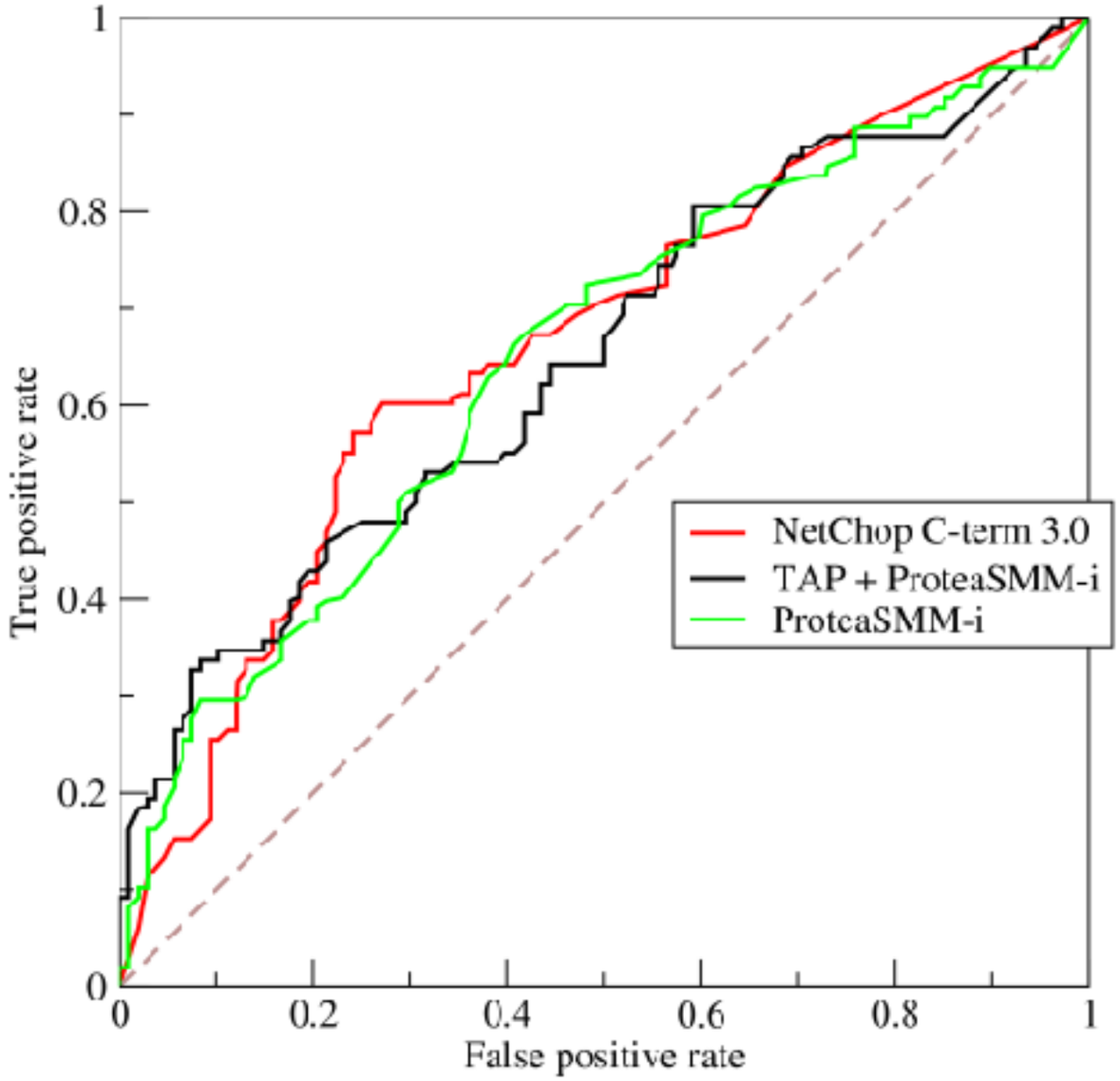
Matriz de Confusión o Tabla de contingencia

$$P_D = \frac{TP}{TP + FN} \quad \text{tasa de detección}$$

$$P_{FA} = \frac{FP}{FP + TN} \quad \text{tasa de falsas alarmas}$$

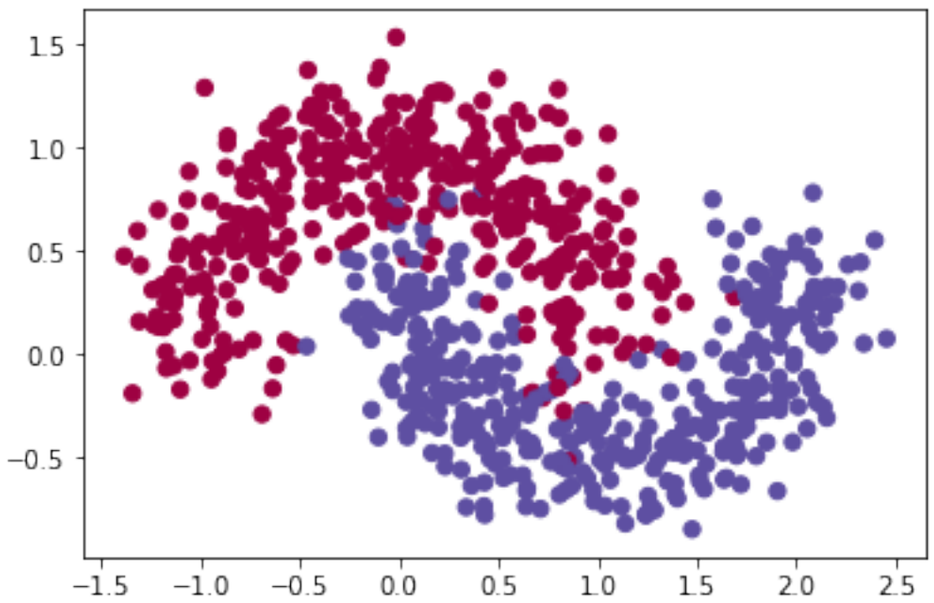
Graficando ROCs puedo comparar la discriminabilidad para comparar efectividad de tratamientos distintos.

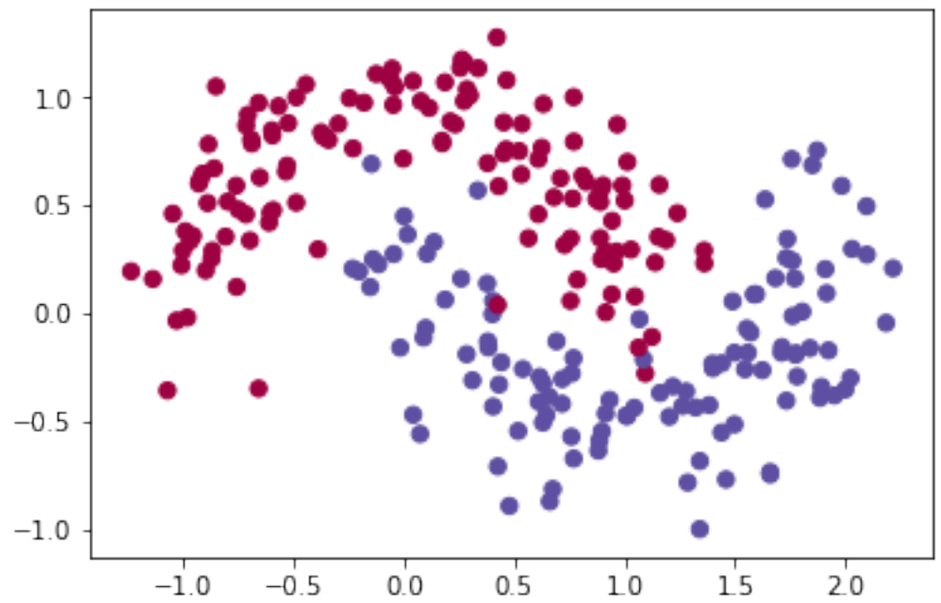
COMPARACIÓN DE ROCS



Curvas ROC empíricas de pruebas diagnósticas distintas con área bajo la curva similares pero comportamiento distinto.

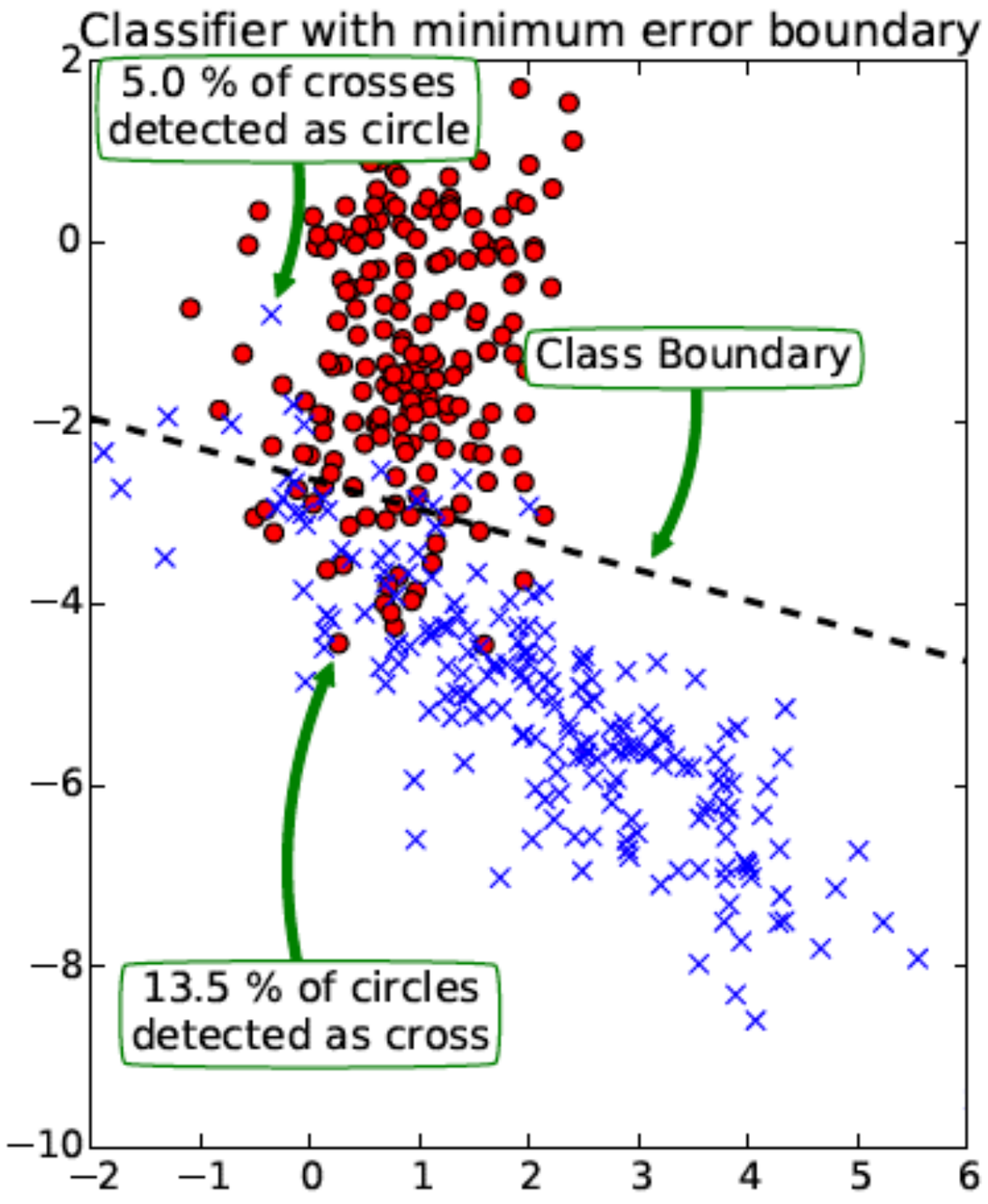
EJEMPLO: ROC Y AUC PROBLEMA DE CLASIFICACIÓN





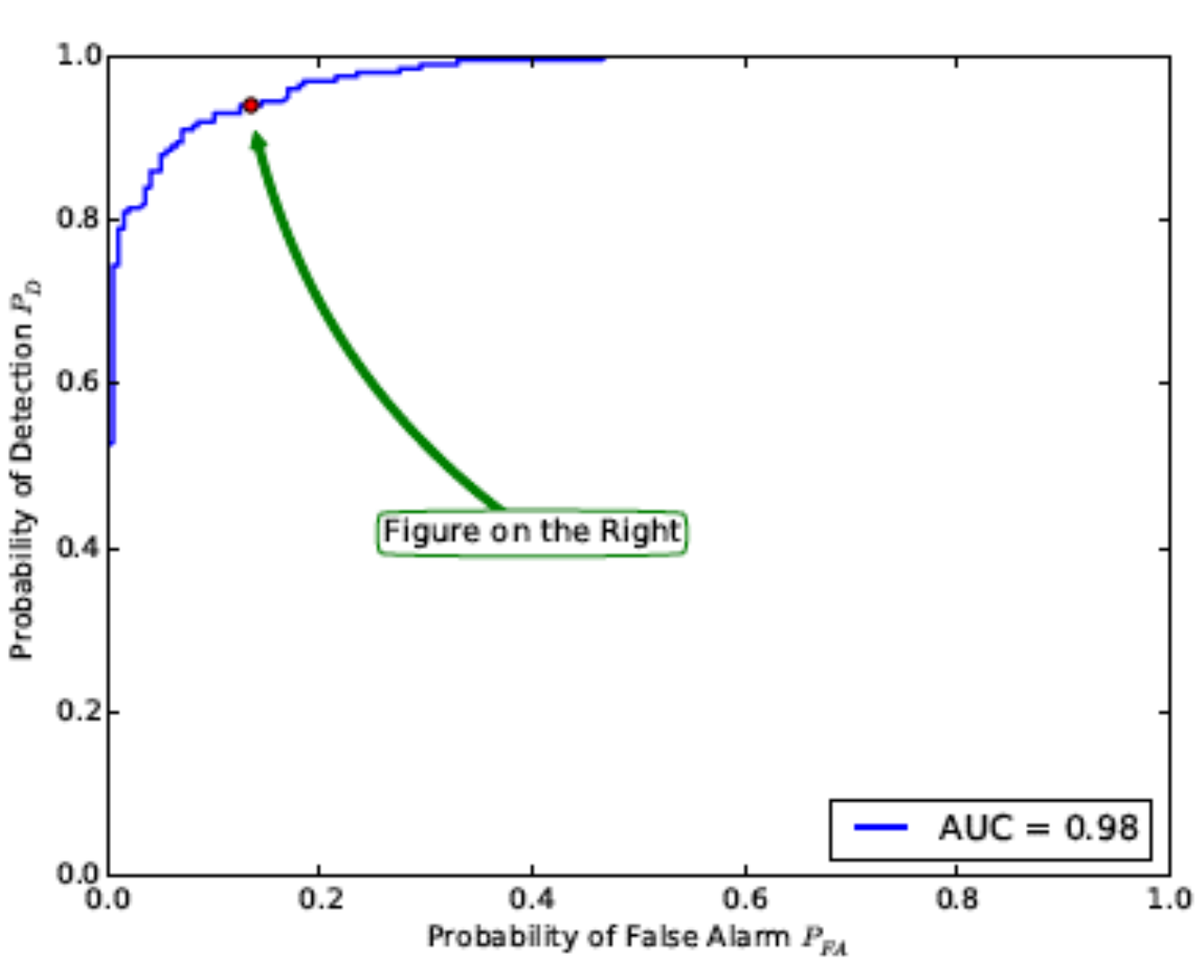
EJEMPLO: ROC Y AUC PROBLEMA DE CLASIFICACIÓN

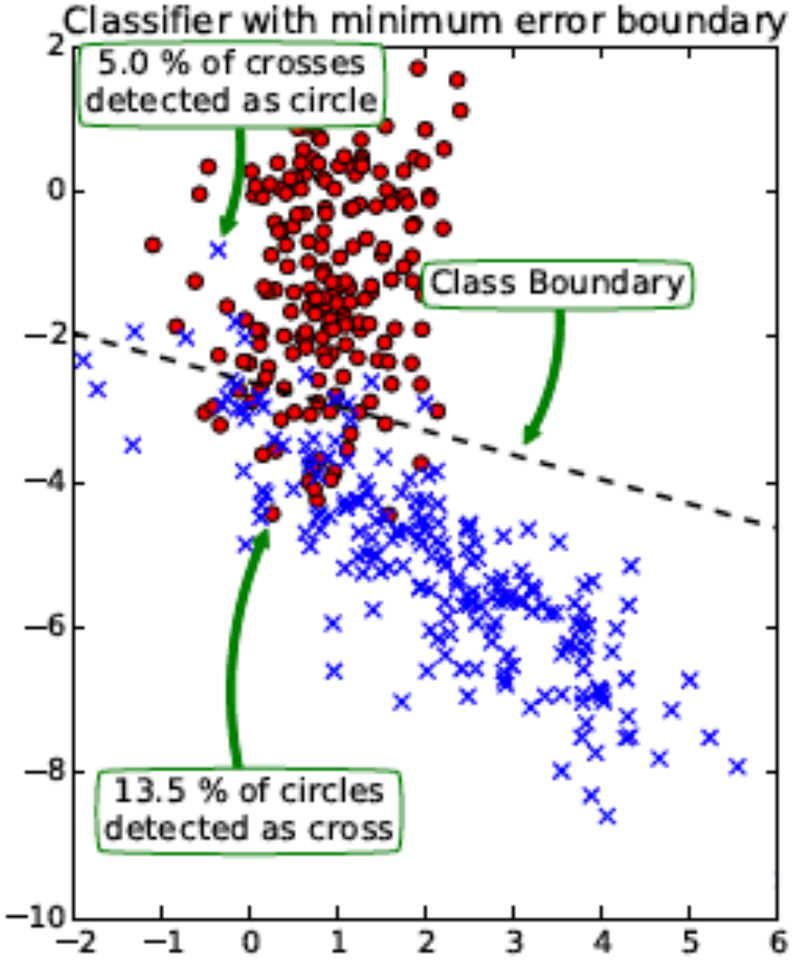
- Se busca un clasificador lineal que minimice el error sobre los datos de entrenamiento.
- Determino los coeficientes de una recta $y=ax+b$, donde a y b se aprenden de los datos de entrenamiento.



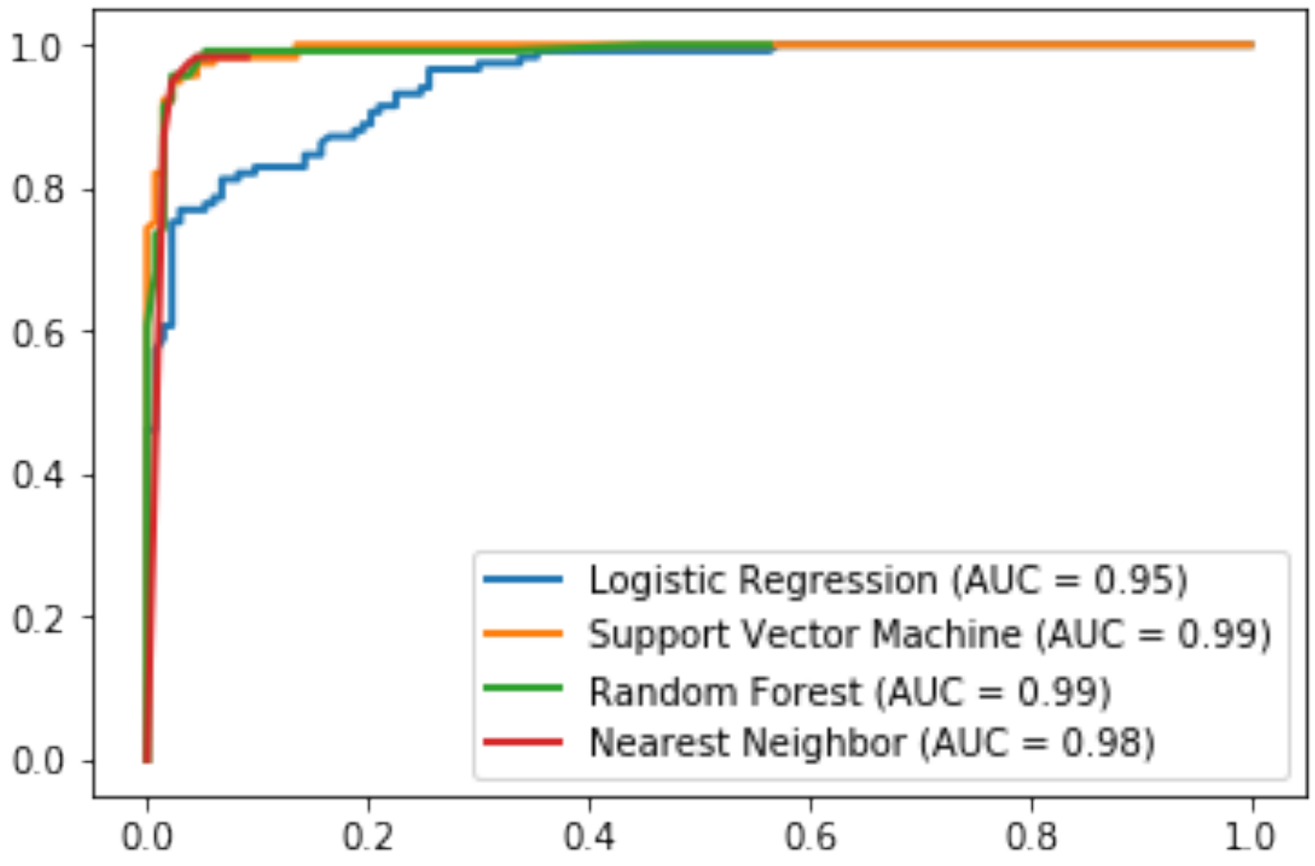
EJEMPLO: ROC Y AUC PROBLEMA DE CLASIFICACIÓN

- Se puede cambiar la sensibilidad del clasificador cambiando los parámetros del clasificador, lo que es una forma de determinar en forma empírica la ROC.





EJEMPLO ROC Y AUC 4 CLASIFICADORES



ROC Y AUC EN PYTHON

```
classifiers = [(LogisticRegression(), "Logistic Regression"),
               (SVC(probability = True), "Support Vector Machine"),
               (RandomForestClassifier(n_estimators = 100), "Random Forest"),
               (KNeighborsClassifier(), "Nearest Neighbor")]

for clf, name in classifiers:
    clf.fit(X, y)

    ROC = []
    for gamma in np.linspace(0, 1, 1000):

        err1 = np.count_nonzero(clf.predict_proba(X_test[y_test == 0, :])[:,1] <= gamma)
        err2 = np.count_nonzero(clf.predict_proba(X_test[y_test == 1, :])[:,1] > gamma)

        err1 = float(err1) / np.count_nonzero(y_test == 0)
        err2 = float(err2) / np.count_nonzero(y_test == 1)

        ROC.append([err1, err2])
    ROC = np.array(ROC)

    ROC = ROC[::-1, :]
    auc = roc_auc_score(y_test, clf.predict_proba(X_test)[:,1])

    plt.plot(1-ROC[:, 0], ROC[:, 1], linewidth = 2, label="%s (AUC = %.2f)" % (name, auc))
```

COMPARACIÓN DE ROCS Y AUCS

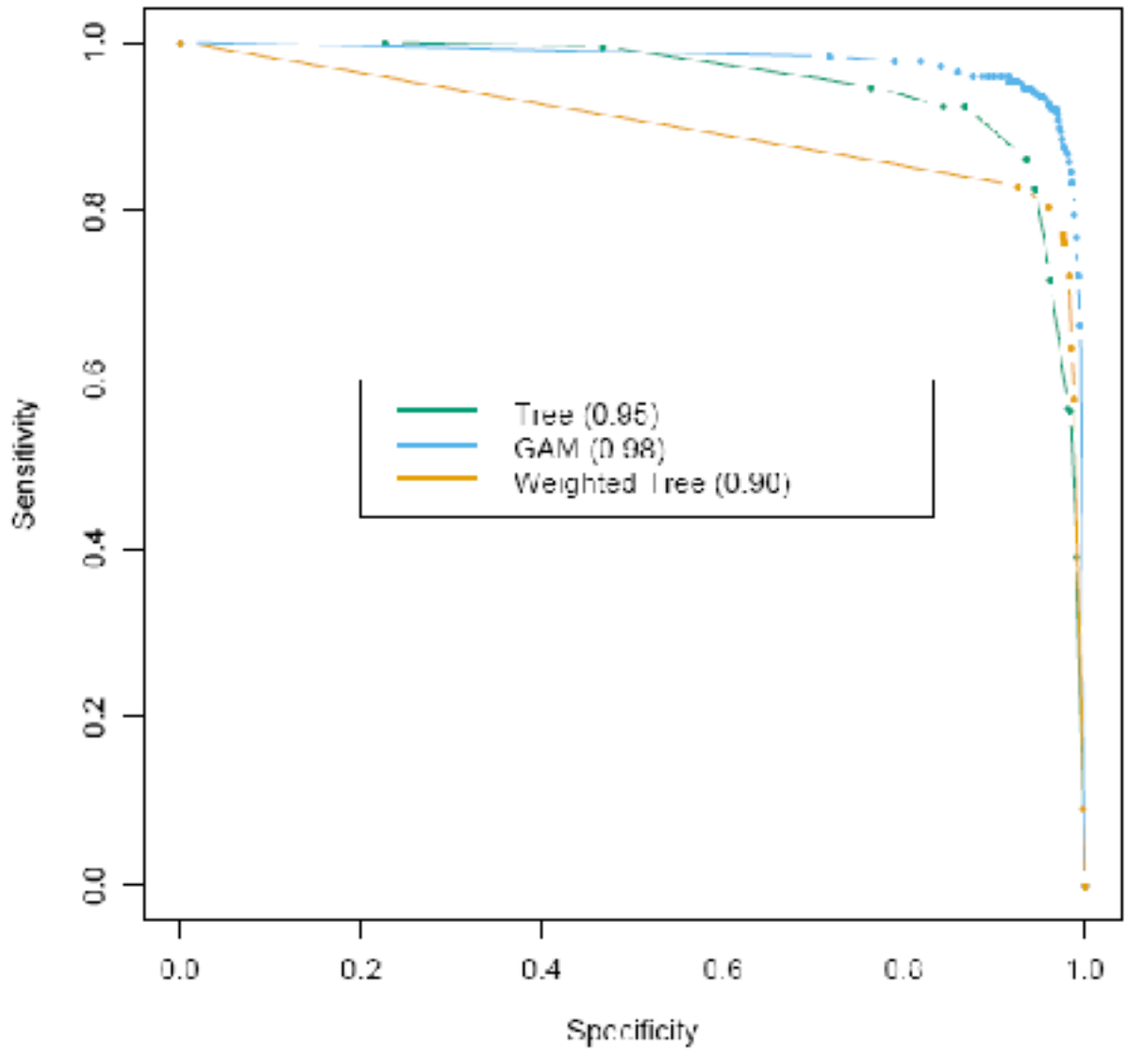


Imagen: Hastie et al.

Comparación de algoritmos de clasificación distintos

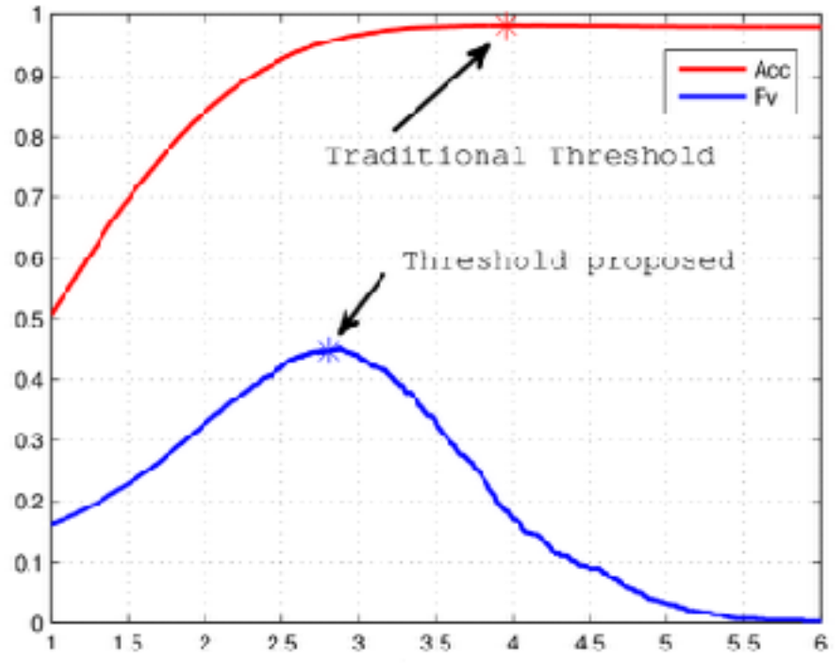
OTRAS MEDIDAS DE DESEMPEÑO

Accuracy: $$A = \frac{TP + TN}{TP + TN + FP + FN}$$

Recall: $$R = \frac{TP}{TP + FN}$$

Precision: $$P = \frac{TP}{TP + FP}$$

F - value: $$Fv = \frac{(1 + \beta^2)RP}{\beta^2 P + R}$$



156	844
125	49875

573	427
1868	48132

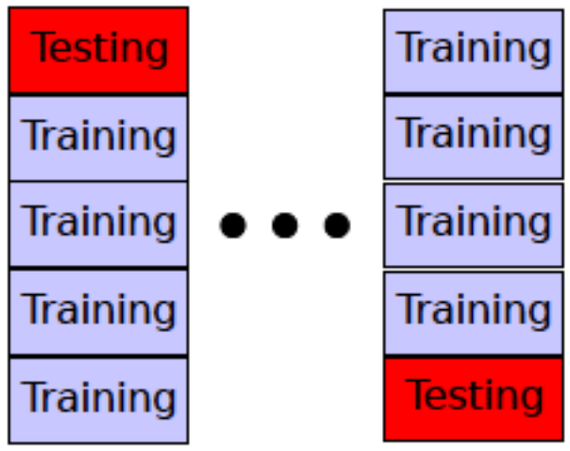
OTRAS MEDIDAS DE DESEMPEÑO

```
# Code:

metrics = ['accuracy',
           'roc_auc',
           'recall',
           'precision',
           'f1']

for scorer in metrics:
    scores = cross_val_score(clf,
                     X,
                     y,
                     scoring = scorer)

    print ("%s score: %.2f +- %.2f" % \
           (scorer,
            np.mean(scores),
            np.std(scores)))
```

```
# Result:

accuracy score: 0.92 +- 0.01
roc_auc score: 0.98 +- 0.00
recall score: 0.90 +- 0.03
precision score: 0.93 +- 0.01
f1 score: 0.91 +- 0.02
```

GENERALIZACIÓN – VALIDACIÓN CRUZADA

- Lo importante es cuán bien clasifica a los datos no vistos.
- Un clasificador sobre entrenado memoriza los datos de entrenamiento y no generaliza
- La capacidad de generalizar debe ser probada con datos no vistos.
- Una forma estándar es hacer k -validación cruzada: 1) separar el conjunto de datos en k subconjuntos. 2)Uso cada conjunto para evaluar mientras que entreno con los restantes. 3) El error lo estimo como la media el desempeño de todos los conjuntos de test. Un valor usual de k es 5 o 10.



VALIDACIÓN CRUZADA

```
from sklearn.model_selection import cross_val_score
scores = cross_val_score(clf, X, y)
```

- La función puede recibir más parámetros, cambiar el número de folds, estratificación, balance.

Code:

```
# Cross-validate to estimate accuracy

scores = cross_val_score(clf,
                         X,
                         y,
                         cv = 10,
                         n_jobs = 2)

for fold in range(10):
    print "Fold %d score: %.2f %%" % \
          (fold, 100*scores[fold])

print "-" * 10
print "CV accuracy: %.2f %%" % \
      (100*np.mean(scores))
```

Result:

```
Fold 0 score: 95.65 %
Fold 1 score: 100.00 %
Fold 2 score: 100.00 %
Fold 3 score: 100.00 %
Fold 4 score: 86.36 %
Fold 5 score: 100.00 %
Fold 6 score: 100.00 %
Fold 7 score: 100.00 %
Fold 8 score: 100.00 %
Fold 9 score: 90.48 %
----------
CV accuracy: 97.25 %
```

VALIDACIÓN CRUZADA ESTRATIFICADA

- En lugar de hacer una partición aleatoria, cada subconjunto tiene la misma proporción de muestras de cada clase que el conjunto original.

```
>>> from sklearn.model_selection import StratifiedKFold
# Generate SKF split. This needs the class labels y.
>>> skf = StratifiedKFold(y, 10)

# skf is a generator; we can transform it into a list.
>>> print list(skf)[0]
(array([0, 1, 4, ... , 399]), array([2, 3, 10, ... , 390]))
# Contains 10 train-test index pairs, above the first one.
# Each index set has same proportion of samples from each class.
```

```
# CV score estimation
from sklearn.model_selection import
     cross_val_score, StratifiedKFold

skf = StratifiedKFold(y, 10, shuffle = True)
scores = cross_val_score(clf, X, y,
                         cv = skf)
```

```
# Print scores

>>> print ("Accuracy: %.2f +- %.2f" %
          (np.mean(scores),
           np.std(scores)))

Accuracy: 0.91 +- 0.04
```

DEJAR UNO AFUERA (LEAVE ONE OUT)

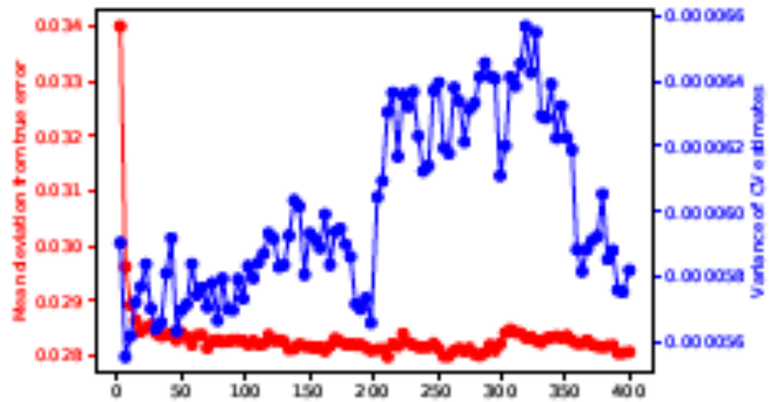
- Caso extremo k fold -cross validation. Se usa cuando tengo pocas muestras. Costoso y con gran varianza.

```
# CV score estimation
from sklearn.model_selection import LeaveOneOut

loo = LeaveOneOut(y.size)
scores = cross_val_score(clf, X, y,
                         cv = loo)
```

```
# Print scores

>>> print ("Accuracy: %.2f +- %.2f" %
           (np.mean(scores),
            np.std(scores)))

Accuracy: 0.92 +- 0.28
```

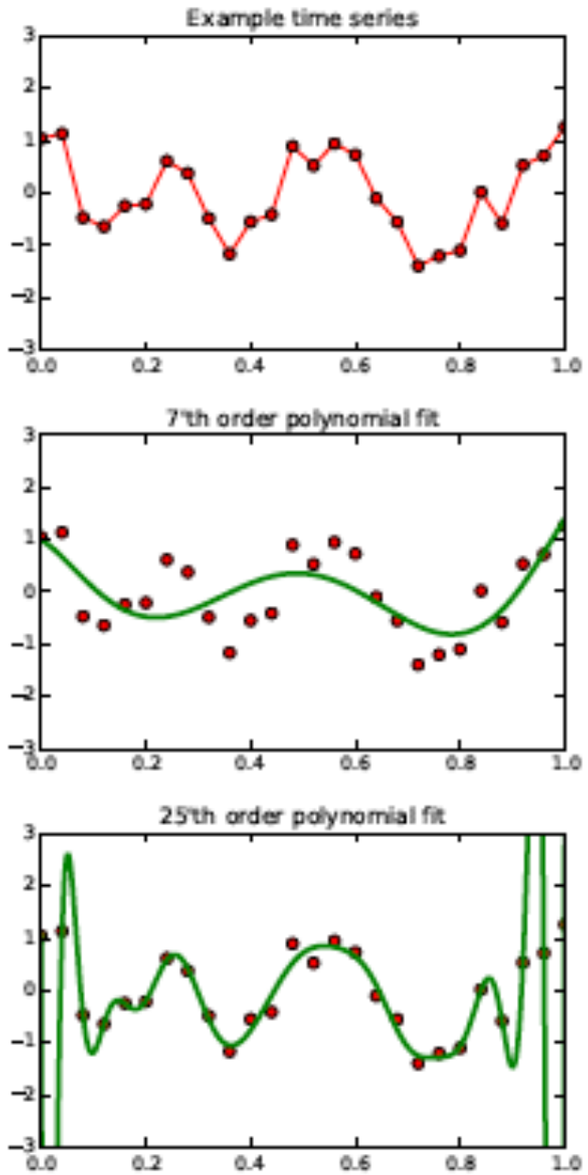
SELECCIÓN DE K

- El mejor k depende de la cantidad de datos, la dificultad del problema, la métrica.
- Cuantos más folds mejor pero la varianza crece cuando se acerca a leave one out.



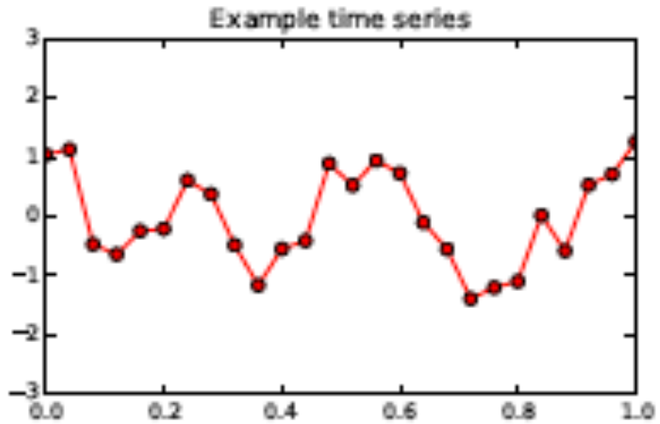
OVERFITTING

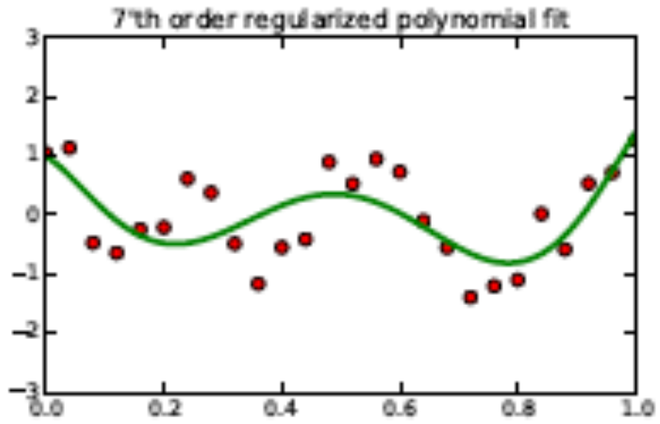
- La generalización está relacionada con el overfitting.
- Los modelos de mayor orden se ajustan mucho a los datos de entrenamiento incluido el ruido de los mismos y por lo tanto pueden ser inestables y con dificultades para generalizar.

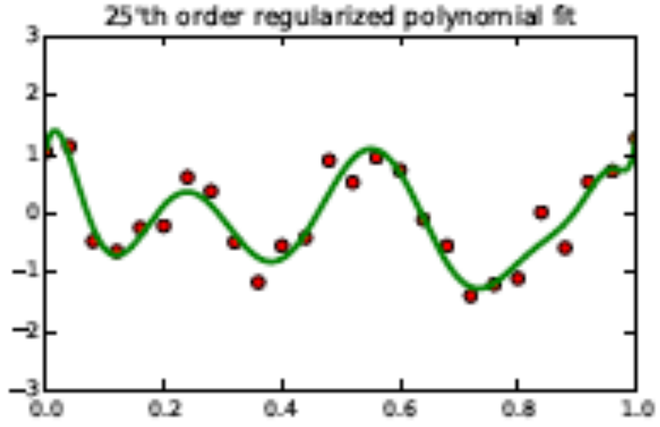


REGULARIZACIÓN

- Una forma de mitigar este efecto es mediante estrategias de regularización que agregan términos de penalidad al error de ajuste.
- Se estimula al modelo para que elija coeficiente pequeños.
- Los coeficientes grandes son costosos.
- Se logran modelos de orden altos, con capacidad de expresión pero que no sigan el ruido de los patrones.







BAYESIANOS- FRECUENTISTAS

- Frecuentistas: aplican Neyman-Pearson: Maximiza la probabilidad de detección sujeto a una probabilidad de falsa alarma determinada. No involucra conocimiento a priori.
- Bayesianos determinan las reglas de decisión que minimizan riesgos utilizando priors.
- Discusión epistemológica:
  - Bayesianos: probabilidad indica grado de creencia no tiene porque haber un experimento
  - Frecuentistas: la probabilidad es la frecuencia de ocurrencia de un evento. Tiene que haber datos y experimento.
- La elección de uno u otro enfoque depende del problema. En los sistemas de radar y sonar se utiliza típicamente Neyman-Pearson mientras que en los sistemas de comunicación se utiliza el riesgo Bayesiano.