Clase 03 - Clasificadores probabilisticos

Aprendizaje Automático Aplicado

Agenda

- Errores con costo diferenciado
- Clasificadores probabilísticos
- Thresholds
- Métricas agnósticas a thresholds
- Evaluación de clasificadores probabilísticos
- Calibración de modelos

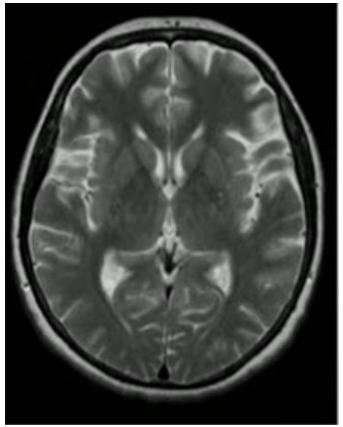
Supongamos un problema de clasificación binaria

tumor / no tumor

15 ejemplos de tumor (10.7%)
125 ejemplo de no tumor (89.3%)

Sistema naive: siempre no tumor

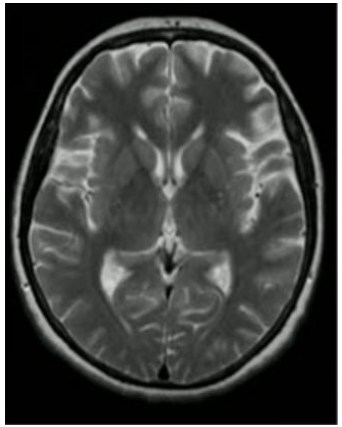
Accuracy: 0.89



Matriz de Confusión

Clase (ground truth)	Decisión: tumor	Decisión: no tumor
tumor	12	3
no tumor	5	120

- Accuracy = (120 + 12) / 140 = 0.943
- Recall = 12 / (12+3) = 0.8
- Precision = 12 / (12 + 5) = 0.7



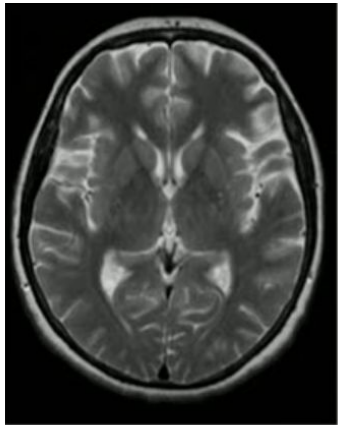
Matriz de Costos

Clase (ground truth)	Decisión: tumor	Decisión: no tumor
tumor	0	10
no tumor	1	0

Definida por expertos en base a conocimiento de dominio.

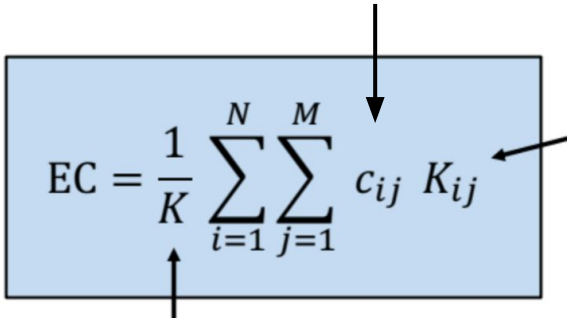
Restricciones:

- Los costos son >= 0
- Todas las clases tienen una mejor desición de costo cero



Costo Esperado

Clase (ground truth) \ Decisión	tumor	no tumor
tumor	0 * 12	10 * 3
no tumor	1 * 5	0 * 120



CE = (0 * 12 + 10 * 3 + 1 * 5 + 0 * 120) / 140 = **0.25**

Tengo una función de costo a medida
Puedo buscar el modelo que la minimiza
Cómo se interpreta?

Costo Esperado Normalizado

$$\text{NaiveEC} = \min_j \sum_{i=1}^{N} c_{ij} P_i$$

Podemos normalizarlo comparándolo con el costo esperado de un sistema *naive*: uno que no tiene acceso a cada instancia y sólo conoce las distribuciones a priori

Porción de ejemplos de la clase *i*

15 ejemplos de tumor (0.107)
125 ejemplo de no tumor (0.893)

Clase (ground truth) \ Decisión	tumor	no tumor
tumor	0 * 0.107	10 * 0.107
no tumor	1 * 0.893	0 * 893
	0.893	1.07

$$\text{NormEC} = \frac{\text{EC}}{\text{NaiveEC}}$$

Queremos que sea < 1

CENorm = 0.25 / 0.893 = 0.279

classifier: predict vs predict_proba

predict: me devuelve una predicción, la clase más probable

predict_proba: me devuelve una distribución de probabilidades sobre todas las clases

Ventajas:

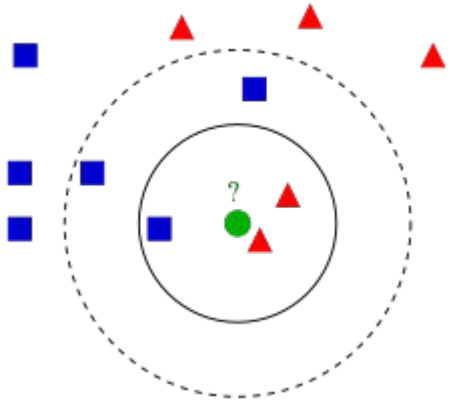
- En algunos casos no quiero tomar la decisión, sino delegarla a un tercero
- Puedo tomar distintas decisiones en diferentes contextos
- Permite tomar otras decisiones (tumor, no tumor, más estudios)

Desventajas:

- No siempre está disponible
- Pueden no ser exactamente probabilidades

predict_proba: ejemplo

- En sklearn.neighbors.KNeighborsClassifier retorna como p(clase) la proporción de vecinos de esa clase que encontró en los k más cercanos



k = 3:
p(azul) = 1/3
p(rojo) = 2/3
k = 5:
p(azul) = 3/5
p(rojo) = 2/5

classifier: predict vs predict_proba

```
>>> print(X_val.shape)
(5, 12)
>>> y_val = clf.predict(X_val)
>>> print(y_val)
[1, 1, 0, 0, 1]
```

```
>>> print(X_val.shape)
(5, 12)
>>> y_val = clf.predict_proba(X_val)
>>> print(y_val)
[[0.1, 0.9],
 [0.2, 0.8],
 [0.8, 0.2],
 [0.9, 0.1],
 [0.0, 1.0]]
>>> y_val[:, 1]
[0.9, 0.8, 0.2, 0.1, 1.0]
```

P(0)

P(1)

Si tomo `y_val > 0.5` para predecir, vuelvo al caso anterior

Thresholds

Puedo tomar un threshold distinto a 0.5 para mis decisiones y obtener mejores métricas, pero:

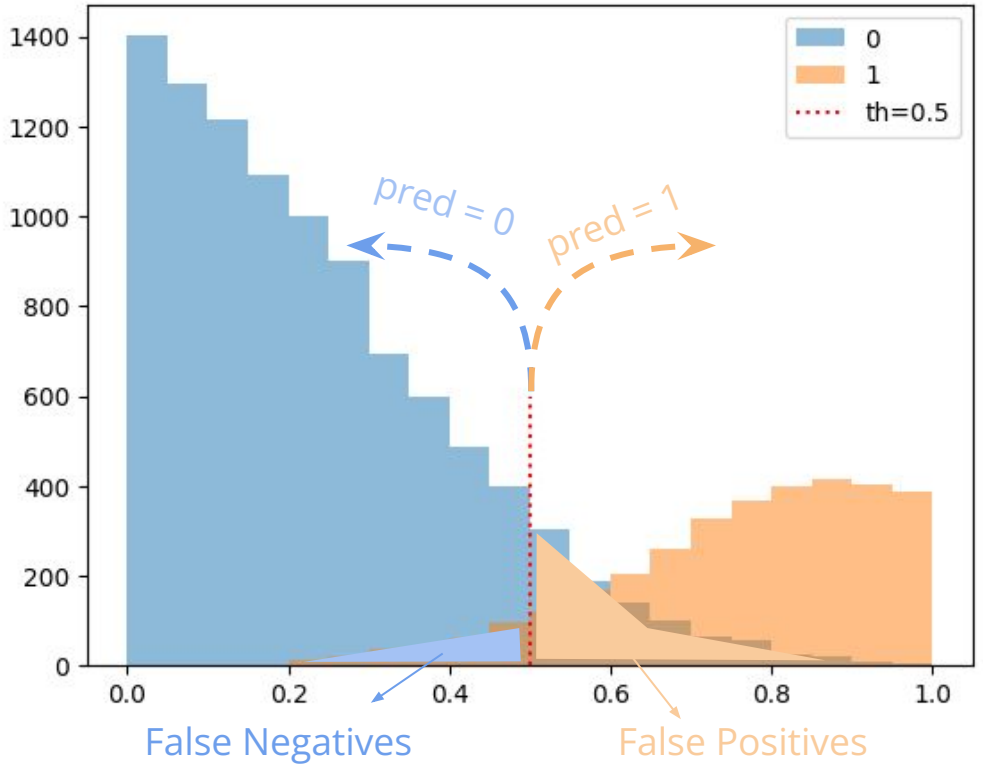
- **No puedo hacerlo en el dataset de train**, porque las probabilidades van a funcionar demasiado bien
- **No puedo hacerlo en test**, porque me voy a sobreajustar
- El dataset en el que lo haga **debe respetar muy bien las proporciones** que voy a tener en producción

Solo tiene sentido cuando estoy haciendo una clasificación

Thresholds

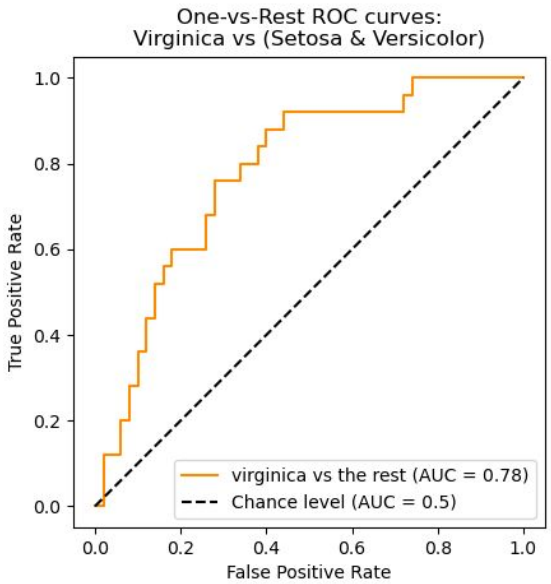
Cómo elegirlo?

- itero valores de thresholds posibles
- En cada uno miro mi métrica
- Me quedo con el threshold que maximiza mi métrica



Métricas agnósticas: auc ROC

ROC: Receiver Operating Characteristic



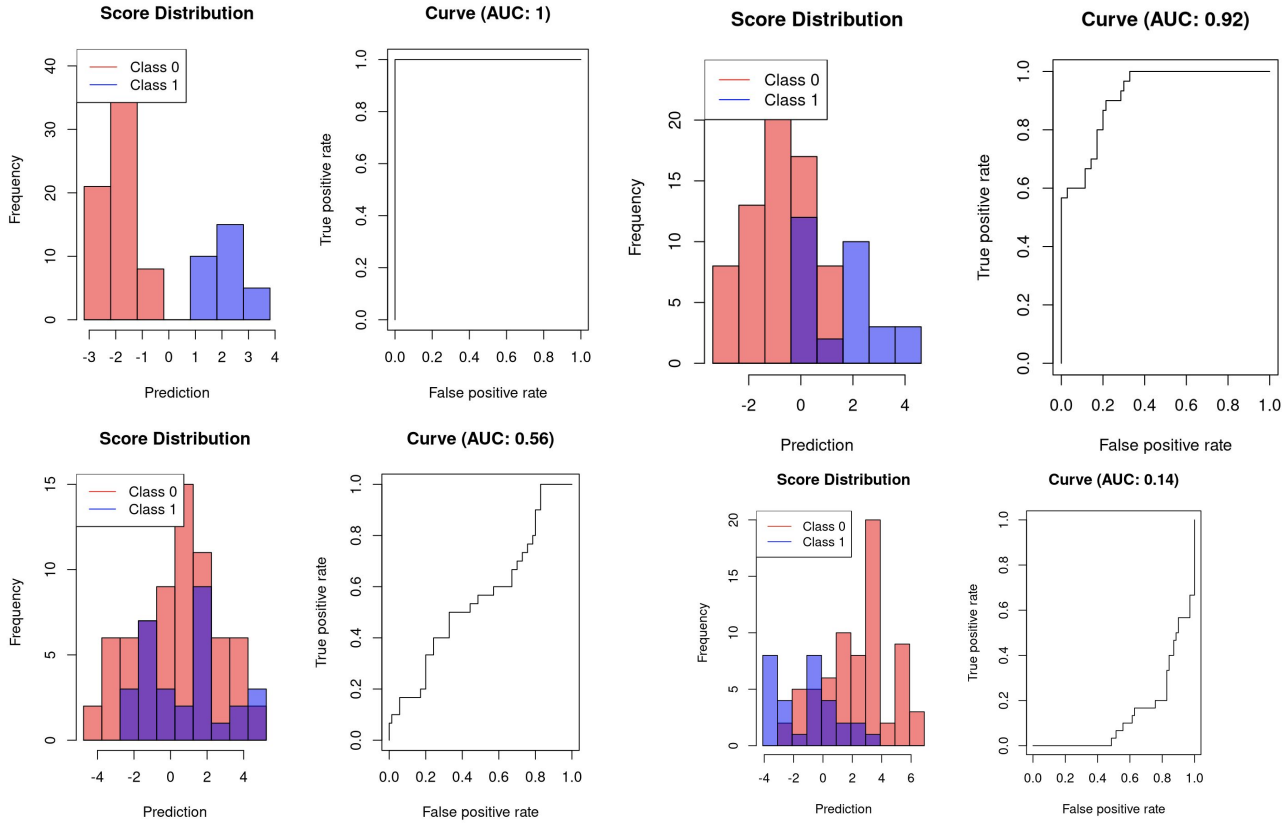
Son medidas que buscan determinar qué tan bien discrimina nuestro clasificador

Medir el área bajo la curva ROC

Interpretable: La probabilidad de que, si tomo un ejemplo positivo x1 y uno negativo x2, al azar, p(x1) > p(x2)

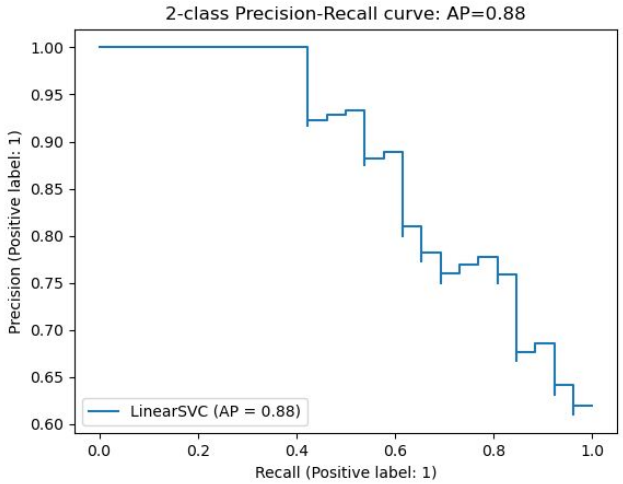
En datasets desbalanceados tiende a sobreestimar

Métricas agnósticas: auc ROC



Métricas agnósticas: auc PR

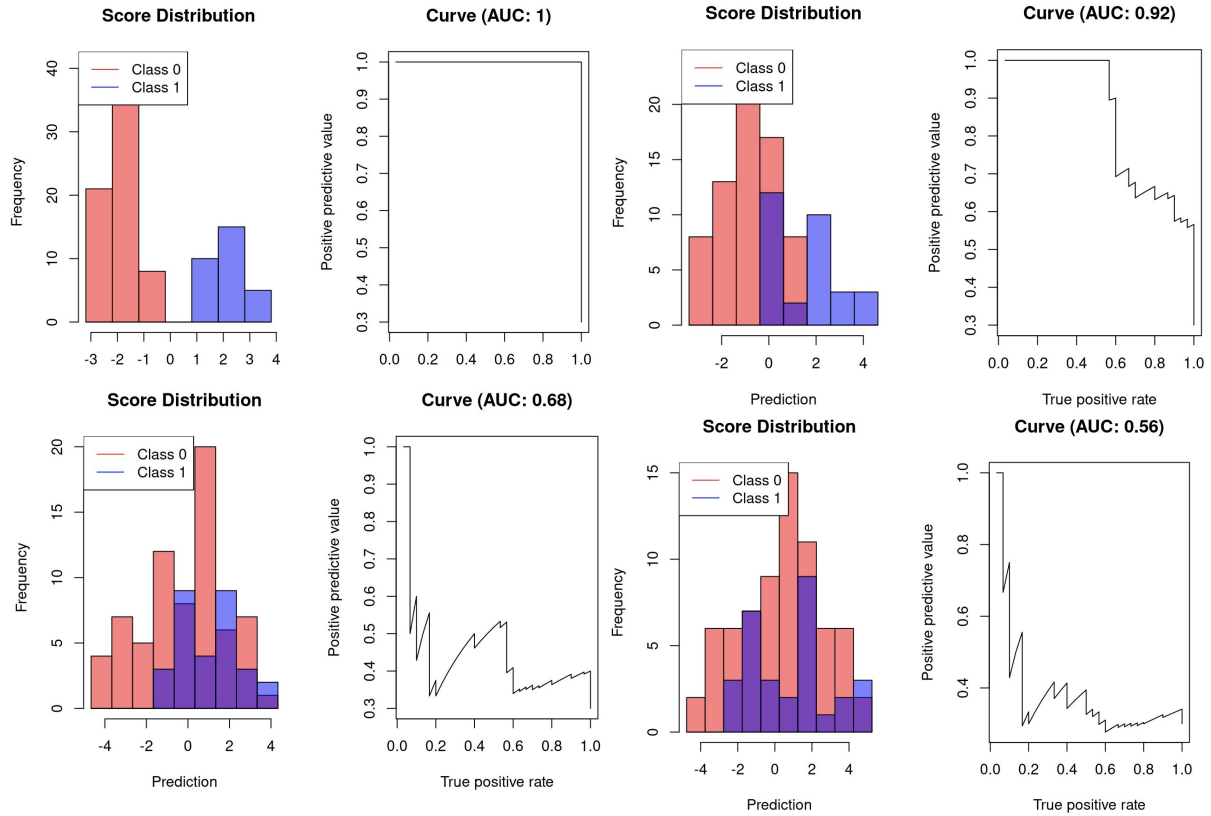
PR: Precision-Recall



Medir el área bajo la curva PR

Funciona bien incluso en datasets desbalanceados

Métricas agnósticas: auc PR



predict_proba: qué quiere decir?

Predicción: sale cara con 50% de probabilidad

Predicción: sale el 4 con 20% de probabilidad

Puedo repetir el evento muchas veces y ver si mi predicción es correcta o no

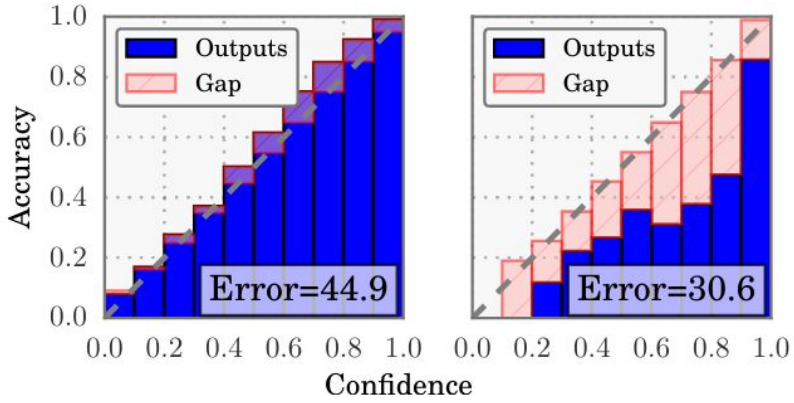
predict_proba: qué quiere decir?

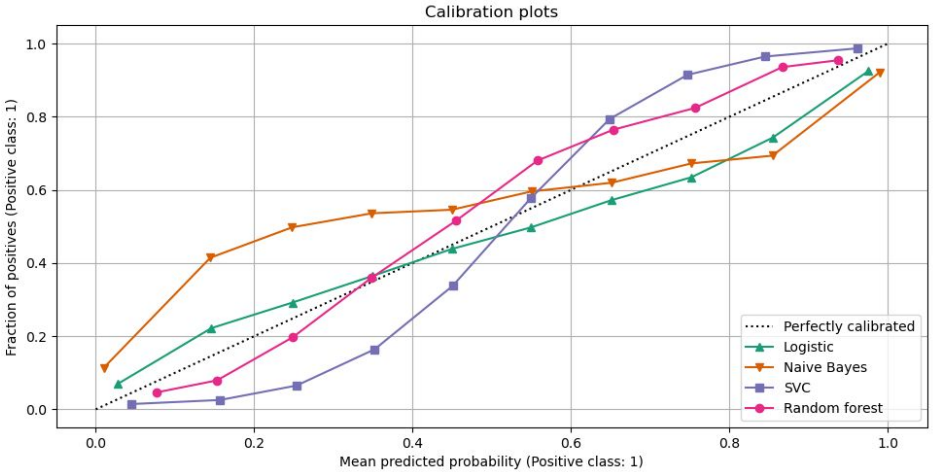
Cómo sé que las predicciones son correctas?

Evento registrado	
Llovió	No Llovió
No Llovió	Llovió

Calibración de modelos

Reliability diagrams





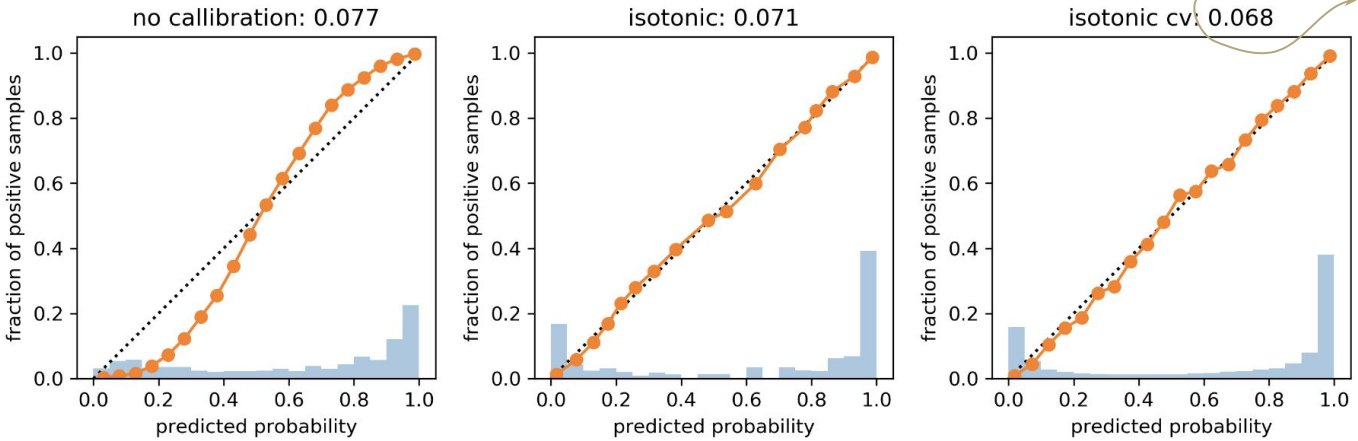
$$\mathrm{ECE} = \sum_{b=1}^{B} \frac{n_b}{N} \left|\mathrm{acc}(b) - \mathrm{conf}(b)\right|,$$

En los casos que predice X% de probabilidad de lluvia, el X% de esos llueve

Calibración de modelos

Si necesito entregar probabilidades
Una vez entrenado puedo **ver si está o no calibrado**
En caso contrario, calibrarlo:
Aprender una función que re-distribuya las probabilidades predichas para que queden calibradas. **No usar training ni testing**

`sklearn.metrics.brier_score_loss`



`sklearn.calibration.CalibratedClassifierCV`

Recap

Si tengo que retornar una probabilidad, **debo verificar que esté calibrada**

Si sólo me interesa la predicción de la clase, puedo elegir un mejor threshold

Que el modelo esté bien calibrado no asegura que sea bueno discriminando, y vice versa

Calibrar un modelo no cambia su capacidad para discriminar

Solo tiene sentido calibrar si necesito retornar probabilidades y mi modelo no está calibrado

Para calibrar o buscar threshold óptimo **nunca hacerlo sobre train ni test**

Referencias

Interpreting ROC Curves, Precision-Recall Curves, and AUCs, blogpost

Machine Learning Fundamentals with a Focus on Evaluation Practices, Luciana Ferrer, slides, video

DECISION THEORY AND CALIBRATION: TWO SIDES OF THE SAME COIN, Luciana Ferrer, slides, video

Probability calibration, scikit-learn user guide