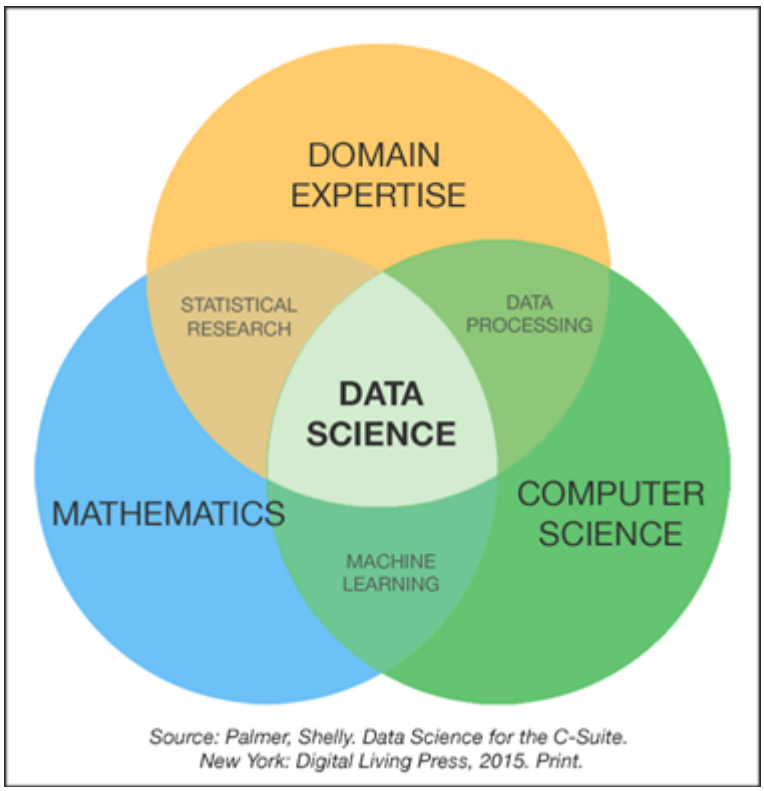
CTIC UNI
CENTRO DE TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIONES
UNIVERSIDAD NACIONAL DE INGENIERÍA

CIENCIA DE DATOS:
APRENDE LOS FUNDAMENTOS DE MANERA PRÁCTICA



SESION 05
APRENDIZAJE SUPERVISADO TECNICAS DE MACHINE LEARNING-PCA

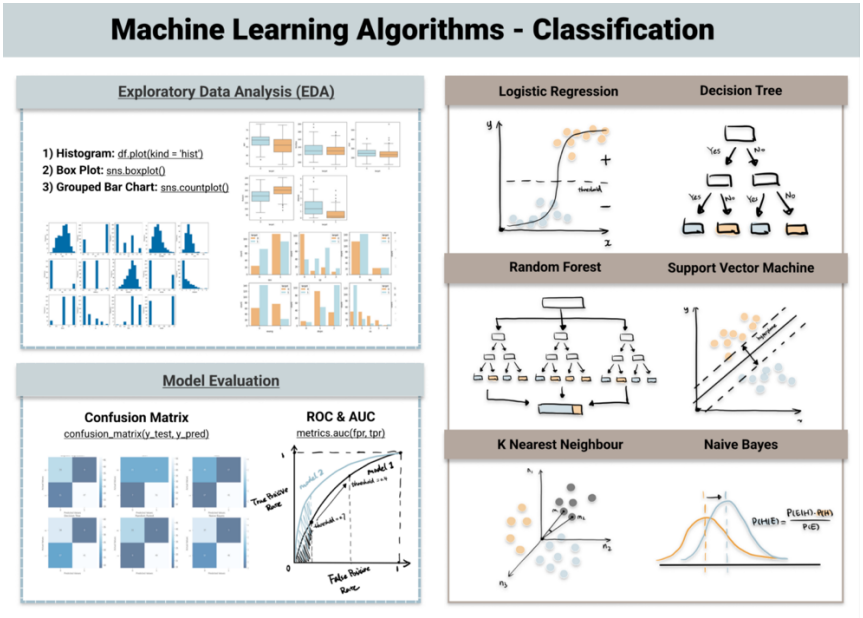
Juan Antonio Chipoco Vidal
jchipoco@gmail.com

ÍNDICE

Objetivo

El objetivo de esta sesion es conocer los algoritmos de machine learning mas utilizados para el aprendizaje supervisado.

Revisaremos como funcionan los algoritmos de clasificacion y luego procederemos a aplicarlos en nuestrar practica semanal. Tambien revisaremos una de las tecnicas de reduccion de dimensionalidad conocida como PCA:



Los principales algoritmos de clasificación en el aprendizaje automático son los siguientes:

Modelos lineales:

Regresión logística
Máquinas de vectores de soporte (SVM)

Modelos no lineales:

K-vecinos más cercanos
Arboles de Decision (Clasificacion)
Bosques Aleatorios (Clasificacion)

Algunos de los problemas en los que se pueden utilizar algoritmos de clasificación son:

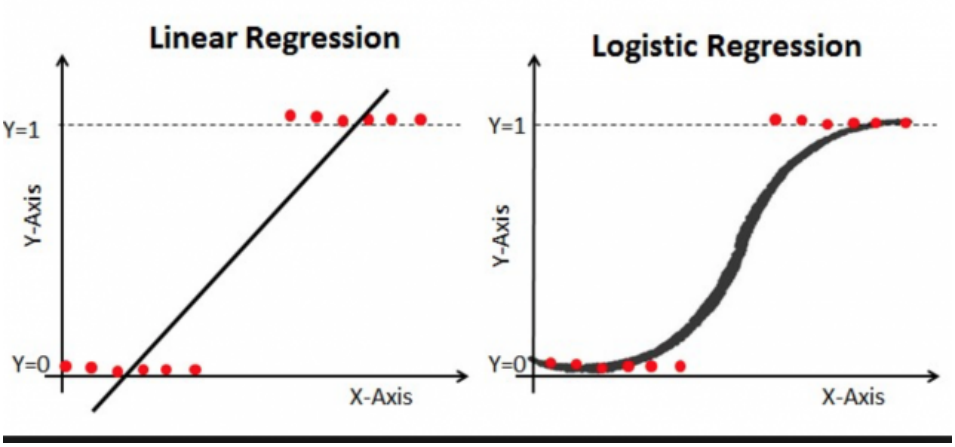
Text Categorization
Fraud Detection
Optical Character Recognition
Face Detection
Language Detection
Customer Segmentation

Regresion Logistica

La regresión logística es un algoritmo de clasificación muy útil en el aprendizaje automático. La regresión logística se utiliza para encontrar una relación entre las entidades de entrada y las etiquetas de salida. Pero eso es lo que hacemos en los problemas de regresión, entonces, ¿cómo es la regresión logística un algoritmo de clasificación?

La respuesta es que en la regresión encontramos el valor de la etiqueta pero se usa la regresión logística para encontrar la categoría de la etiqueta.

Por ejemplo, si queremos predecir las calificaciones de un estudiante, entonces este es el problema de regresión, pero si queremos predecir si el estudiante aprobará o reprobará el examen, entonces es el problema de clasificación donde se puede usar la regresión logística.

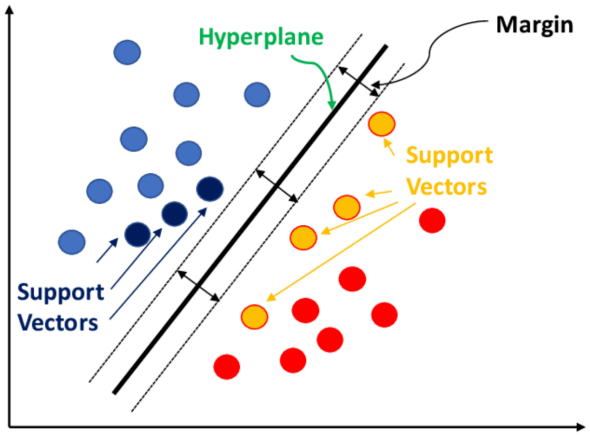


Support Vector Machines (SVM)

Support Vector Machine es un algoritmo de aprendizaje automático muy potente que se puede utilizar tanto para tareas de clasificación como de regresión. Pero dado que SVM se usa más en la clasificación, podemos decir que también es un algoritmo de clasificación.

El algoritmo SVM se utiliza para aprender half-spaces con algún conocimiento de la preferencia de margen grande. Hay dos tipos de SVM; Hard-SVM, Soft-SVM. El Hard-SVM encuentra el half-space que divide perfectamente los datos con el mayor margen. Soft-SVM no encuentra ningún punto de la división, permite violar las restricciones hasta cierto punto.

La gran fortaleza de SVM es que el entrenamiento es muy simple. No requiere ningún óptimo, a diferencia de las redes neuronales. También ajusta muy bien los datos a datos dimensionales muy altos.

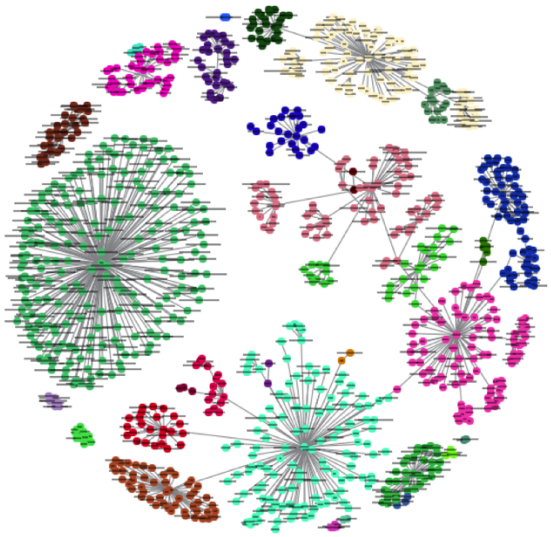


K Nearest Neigbhor

K Nearest Neighbor es uno de los algoritmos de clasificación más simples en el aprendizaje automático. Funciona aprendiendo de los datos de entrenamiento y luego predice la etiqueta de cualquier categoría en función de las etiquetas de sus vecinos más cercanos en los datos de entrenamiento.

De ello se deduce que las características utilizadas para describir la estructura de los puntos de datos son más relevantes para sus etiquetas de una manera que los acerca a los puntos para tener la misma etiqueta.

KNN es un algoritmo de clasificación de aprendizaje automático muy simple que se basa en la suposición de que los elementos que se parecen deben ser iguales. Necesitamos que todo el conjunto de entrenamiento se almacene durante el entrenamiento, y mientras probamos el algoritmo, necesitamos probar todo el conjunto de datos para encontrar el vecino más cercano.



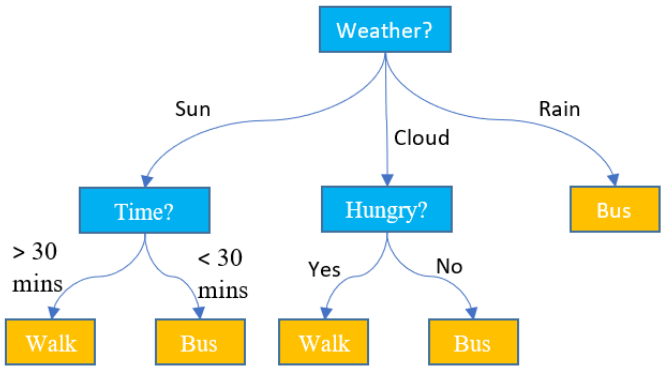
Decision Tree

Un árbol de decisión es un algoritmo que predice la etiqueta asociada con una instancia viajando desde un nodo raíz de un árbol hasta una hoja. Por ejemplo, necesitamos clasificar si la papaya es sabrosa o no, veamos cómo se expresará el árbol de decisión en este problema.

Para clasificar si la papaya es sabrosa o no, el algoritmo del árbol de decisión verificará primero el color de la papaya. Si el color no se encuentra entre el verde pálido y el amarillo pálido, entonces el algoritmo predecirá que la papaya no es sabrosa sin observar más características.

Entonces, lo que sucede es que comenzamos con un árbol con una sola hoja y lo etiquetamos según la mayoría de las etiquetas que tenemos en el conjunto de aprendizaje. Luego hacemos algunas iteraciones, con cada iteración vemos el efecto de dividir la hoja única.

Luego, entre todo el número posible de divisiones, o seleccionamos la que maximiza la ganancia, o elegimos no seleccionar la hoja en absoluto.

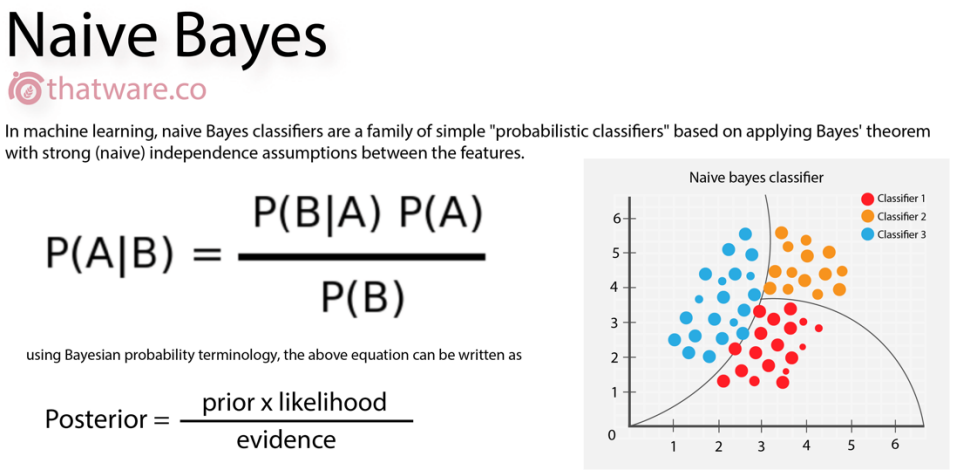


Naive Bayes

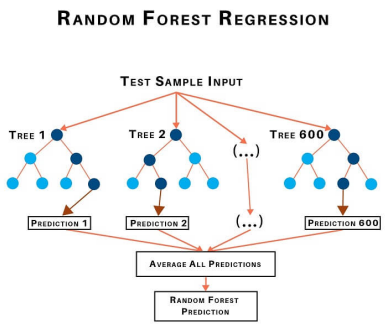
Naive Bayes es un poderoso algoritmo de aprendizaje automático supervisado que se utiliza para problemas de clasificación. Utiliza características para predecir una variable objetivo. La diferencia entre Naive Bayes y otros algoritmos de clasificación es que Naive Bayes asume que las características son independientes entre sí y que no hay correlación entre las características.

Entonces lo que pasa es que esta hipótesis no se evalúa en base a cuestiones de la vida real. Por lo tanto, esta suposición ingenua de que las características no están correlacionadas es la razón por la que este algoritmo se conoce como Naive Bayes.

Este algoritmo es una demostración clásica de cómo las suposiciones generativas y las estimaciones de parámetros pueden simplificar el proceso de aprendizaje. En el algoritmo Naive Bayes, hacemos predicciones asumiendo que las características dadas son independientes entre sí.



Random Forest Classifier



Recordemos que la varianza es la medida de la variabilidad de un conjunto de datos que indica hasta qué punto se distribuyen los diferentes valores. Matemáticamente, se define como la suma de los cuadrados de las diferencias entre una variable y su media, dividido entre el numero de datos.

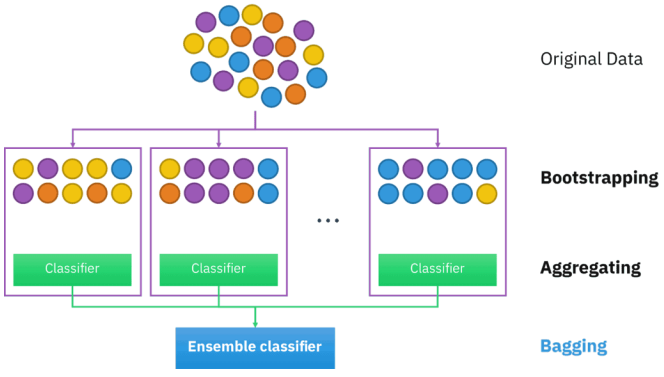
Veamos el caso de los árboles de decisión. Como sabemos, pueden reconstruir patrones muy complejos, pero tienden a tener un rendimiento inferior incluso si se producen cambios menores en los datos. Es por eso que un árbol de decisiones independiente no obtendrá grandes resultados. Aún así, si compone muchos de estos árboles, el rendimiento predictivo mejorará drásticamente. Esto es un método de conjunto llamado Random Forest.

¿Qué es el esemble learning?

La idea general del *esemble learning* es bastante simple. Debe entrenar varios algoritmos de ML y combinar sus predicciones de alguna manera. Tal enfoque tiende a hacer predicciones más precisas que cualquier modelo individual. Un modelo Ensemble es un modelo que consta de muchos modelos base.

Bagging

Bootstrap Aggregating o Bagging es una técnica bastante simple pero realmente poderosa. Comprender el concepto general de Bagging es realmente crucial, ya que es la base del algoritmo Random Forest (RF).



Random Forest Classifier

En general, el bagging es una buena técnica que ayuda a manejar el sobreajuste y reduce la varianza.

¿Qué es un bosque aleatorio?

Random Forest es un algoritmo de aprendizaje supervisado que se basa en el método de esemble learning y muchos árboles de decisión. Random Forest es una técnica de bagging, por lo que todos los cálculos se ejecutan en paralelo y no hay interacción entre los árboles de decisión al construirlos. RF se puede utilizar para resolver tareas de clasificación y regresión.

El nombre "bosque aleatorio" proviene de la idea de bagging de la aleatorización de datos (aleatorio) y la construcción de múltiples árboles de decisión (bosque). En general, es un poderoso algoritmo de ML que limita las desventajas de un modelo de árbol de decisiones.

Métricas de performance para clasificacion

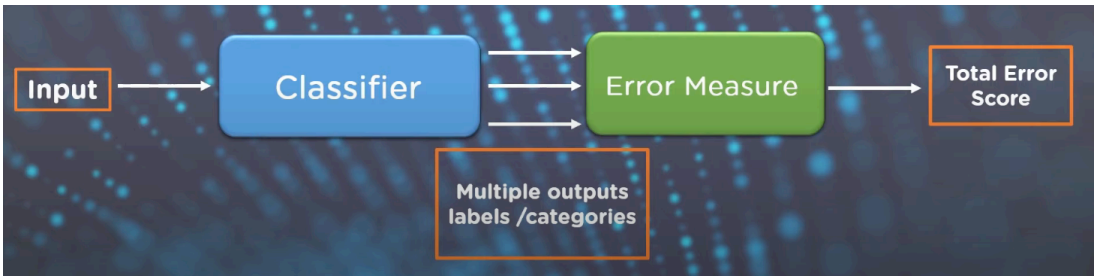
Metric	Formula	Evaluation focus
Accuracy	$ACC = \frac{TP + TN}{TP + TN + FP + FN}$	Overall effectiveness of a classifier
Precision	$PRC = \frac{TP}{TP + FP}$	Class agreement of the data labels with the positive labels given by the classifier
Sensitivity	$SNS = \frac{TP}{TP + FN}$	Effectiveness of a classifier to identify positive labels. Also called true positive rate (TPR)
Specificity	$SPC = \frac{TN}{TN + FP}$	How effectively a classifier identifies negative labels. Also called true negative rate (TNR)
F_1 score	$F_1 = 2\frac{PRC \cdot SNS}{PRC + SNS}$	Combination of precision (PRC) and sensitivity (SNS) in a single metric
Geometric mean	$GM = \sqrt{SNS \cdot SPC}$	Combination of sensitivity (SNS) and specificity (SPC) in a single metric
Area under (ROC) curve	$AUC = \int_0^1 SNS \cdot dSPC$	Combined metric based on the receiver operating characteristic (ROC) space (*Powers, 2011*)

- Los problemas de clasificación son una de las áreas más investigadas del mundo. Los casos de uso están presentes en casi todos los entornos industriales y de producción. Reconocimiento de voz, reconocimiento facial, clasificación de texto: la lista es interminable.
- Los modelos de clasificación tienen una salida discreta, por lo que necesitamos una métrica que compare clases discretas de alguna forma. Las métricas de clasificación evalúan el rendimiento de un modelo y te dicen qué tan buena o mala es la clasificación, pero cada una de ellas lo evalúa de manera diferente.
- Para evaluar los modelos de clasificación, discutiremos estas métricas en detalle:
- Matriz de confusión

  - ✓ Accuracy
  - ✓ Precisión
  - ✓ Recall (Sensitivity)
  - ✓ F1-Score
  - ✓ ROC-AUC

Matriz de confusion

- Los modelos de clasificacion tienen multiples categorias de salida. La mayoria de las metricas de error nos indicaran el error total de un modelo, pero a partir de eso no podremos averiguar errores individuales en nuestro modelo.

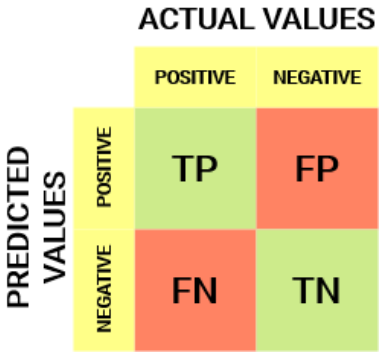


- Una matriz de confusión es una matriz N x N utilizada para evaluar el rendimiento de un modelo de clasificación, donde N es el número de clases objetivo. La matriz compara los valores objetivo reales con los predichos por el modelo de aprendizaje automático. Esto nos da una visión holistica de qué tan bien está funcionando nuestro modelo de clasificación y qué tipo de errores está cometiendo.

		Predicted Class: Positive	Predicted Class: Negative	
Actual Class	Positive	True Positive (TP)	False Negative (FN) **Type II Error**	**Sensitivity** $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) **Type I Error**	True Negative (TN)	**Specificity** $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	**Negative Predictive Value** $\frac{TN}{(TN+FN)}$	**Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$

Matriz de confusion

- Analizemos para el caso de una clasificacion binaria, supongamos que deseamos predecir cuántas personas están infectadas con un virus peligroso antes de que muestren los síntomas y en base al eso aislarlos de la población sana.



True Positive (TP)

El valor predicho coincide con el valor real.

El valor real fue positivo y el modelo predijo un valor positivo

True Negative (TN)

El valor predicho coincide con el valor real

El valor real fue negativo y el modelo predijo un valor negativo

False Positive (FP): Type I error

El valor predicho fue predicho falsamente

El valor real fue negativo pero el modelo predijo un valor positivo

También conocido como error tipo 1.

False Negative (FN): Type II error

El valor predicho fue predicho falsamente

El valor real fue positivo pero el modelo predijo un valor negativo

También conocido como el error tipo 2

- Prosiguiendo con nuestro ejemplo nos damos cuenta que nuestro conjunto de datos es un ejemplo de un conjunto de datos no balanceados (*imbalanced dataset*). Hay 997 puntos de datos para la clase negativa y 3 puntos de datos para la clase positiva.

ID	Actual Sick?	Predicted Sick?	Outcome
1	1	1	TP
2	0	0	TN
3	0	0	TN
4	1	1	TP
5	0	0	TN
6	0	0	TN
7	1	0	FP
8	0	1	FN
9	0	0	TN
10	1	0	FP
:	:	:	:
1000	0	0	FN

- **Accuracy,** la exactitud se refiere a cuán cerca del valor real se encuentra el valor medido.

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

De la tabla obtenemos TP = 30, TN = 930, FP = 30, FN = 10

$$Accuracy = \frac{30 + 930}{30 + 30 + 930 + 10} = 0.96$$

El 96% nos indica que porcentaje de casos negativos y positivos acerto el modelo. Nada mas. Esto nos conduce a introducir los conceptos de Precision y Recall.

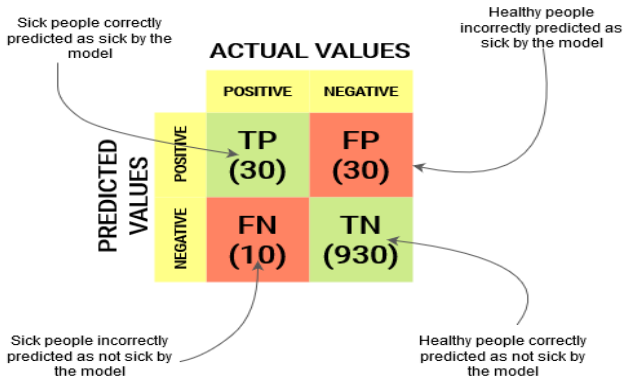
- **Precision**, se refiere a la dispersión del conjunto de valores obtenidos de mediciones repetidas de una magnitud. Cuanto menor es la dispersión mayor la precisión. Nos dice cuántos de los casos predichos como positivos resultaron ser positivos realmente.

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

- **Recall**, la sensitividad (True positive rate) nos dice cuántos de los casos positivos reales pudimos predecir correctamente con nuestro modelo.

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

Matriz de confusion



$$Precision = \frac{30}{30+30} = 0.5$$

$$Recall = \frac{30}{30+10} = 0.75$$

El 50% por ciento de los casos pronosticados correctamente resultaron ser casos positivos. Mientras que el 75% de los positivos fueron predichos con éxito por nuestro modelo.

Precision es una métrica útil en los casos en los que los falsos positivos son más preocupantes que los falsos negativos.

La precisión es importante en los sistemas de recomendación de música o video, sitios web de comercio electrónico, etc. Los resultados incorrectos pueden provocar la pérdida de clientes y ser perjudiciales para el negocio.

Recall es una métrica útil en los casos en que el Falso Negativo supera al Falso Positivo.

Recall es importante en los casos médicos en los que no importa si activamos una falsa alarma, Los casos positivos reales no deben pasar desapercibidos, pues no queremos dar de alta accidentalmente a una persona infectada y dejar que se mezcle con la población sana, propagando así el virus contagioso.

En los casos que no esta claro cual de los dos es mas importante procedemos a combinarlos.

- **F1-Score,** en la práctica, cuando tratamos de aumentar la precisión de nuestro modelo, el recall disminuye y viceversa. La puntuación F1 captura ambas tendencias en un solo valor**:**

$$F1 - score = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}}$$

Matriz de confusion

- **False Negative Rate** (FNR) nos dice qué proporción de la clase positiva fue clasificada incorrectamente por el clasificador.

  Es deseable un TPR más alto y un FNR más bajo, ya que queremos clasificar correctamente la clase positiva.

$$FNR = \frac{FN}{TP + FN}$$

- **Specificity** (**True Negative Rate** , TNR) La especificidad nos dice qué proporción de la clase negativa se clasificó correctamente.

  Tomando el mismo ejemplo que en Sensibilidad, Especificidad significaría determinar la proporción de personas sanas que fueron identificadas correctamente por el modelo.

$$Specificity = \frac{TN}{TN + FP}$$

- **False Positive Rate** (FPR) nos dice qué proporción de la clase negativa fue clasificada incorrectamente por el clasificador.

  Es deseable una TNR más alta y una FPR más baja, ya que queremos clasificar correctamente la clase negativa.

$$FPR = \frac{FP}{TN + FP} = 1 - Specificity$$

- De todas estas métricas, la Sensibilidad y la Especificidad son quizás las más importantes y veremos más adelante cómo se utilizan para construir una métrica de evaluación. Pero antes de eso, entendamos por qué la probabilidad de predicción es mejor que predecir la clase objetivo directamente.
- **Probabilidad de predicciones,** se puede usar un modelo de clasificación de aprendizaje automático para predecir la clase real del punto de datos directamente o predecir su probabilidad de pertenecer a diferentes clases. Esto último nos da más control sobre el resultado. Podemos determinar nuestro propio umbral para interpretar el resultado del clasificador. Establecer diferentes umbrales para clasificar la clase positiva para los puntos de datos cambiará inadvertidamente la Sensibilidad y la Especificidad del modelo. Y uno de estos umbrales probablemente dará un mejor resultado que los demás, dependiendo de si nuestro objetivo es reducir el número de falsos negativos o falsos positivos.

Matriz de confusion

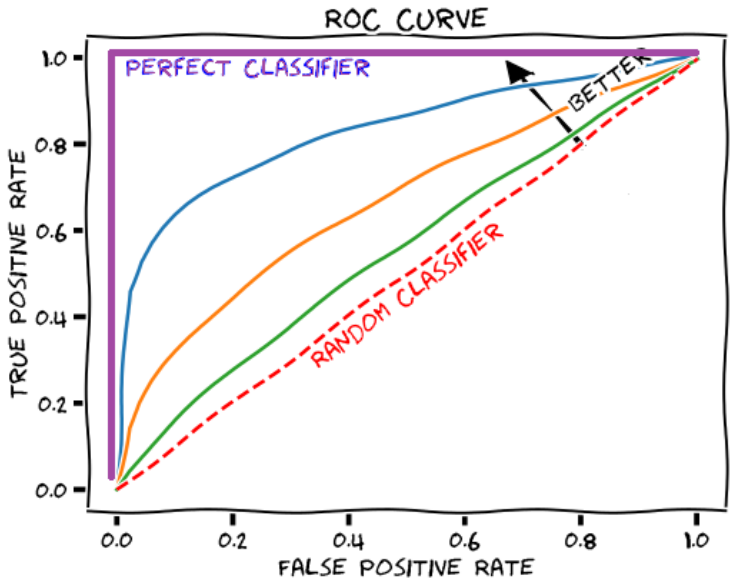
Veamos la siguiente tabla:

ID	Actual	Prediction Probability	>0.6	>0.7	> 0.8	Metric
1	0	0.98	1	1	1	
2	1	0.67	1	0	0	
3	1	0.58	0	0	0	
4	0	0.78	1	1	0	
5	1	0.85	1	1	1	
6	0	0.86	1	1	1	
7	0	0.79	1	1	0	
8	0	0.89	1	1	1	
9	1	0.82	1	1	1	
10	0	0.86	1	1	1	
			0.75	0.5	0.5	TPR
			1	1	0.66	FPR
			0	0	0.33	TNR
			0.25	0.5	0.5	FNR

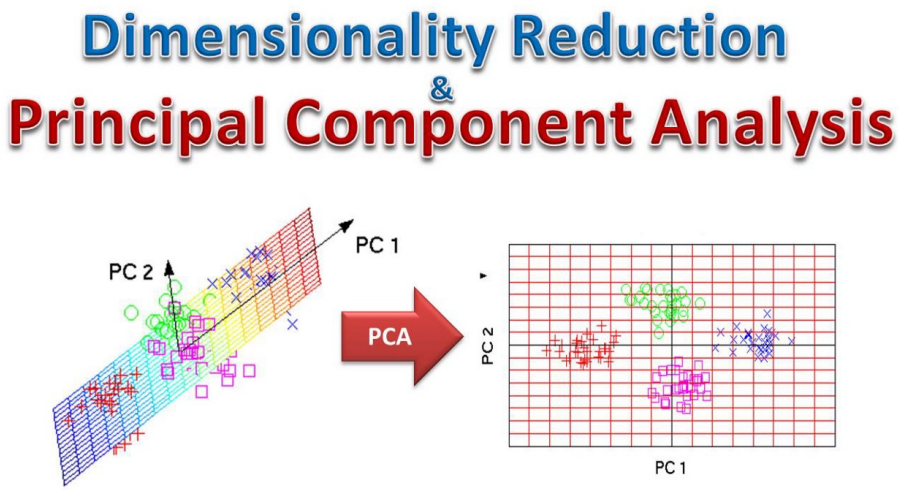
Las métricas cambian con los valores de umbral cambiantes. Podemos generar diferentes matrices de confusión y comparar las diversas métricas que discutimos en la sección anterior. Pero no sería prudente hacerlo. En cambio, lo que podemos hacer es generar un gráfico entre algunas de estas métricas para que podamos visualizar fácilmente qué umbral nos está dando un mejor resultado. Esta es la curva AUC-ROC.

Curva AUC-ROC

La curva característica del operador del receptor (ROC) es una métrica de evaluación para problemas de clasificación binaria. Es una curva de probabilidad que traza la TPR contra la FPR con varios valores de umbral y esencialmente separa la "señal" del "ruido". El área bajo la curva (AUC) es la medida de la capacidad de un clasificador para distinguir entre clases y se utiliza como resumen de la curva ROC.

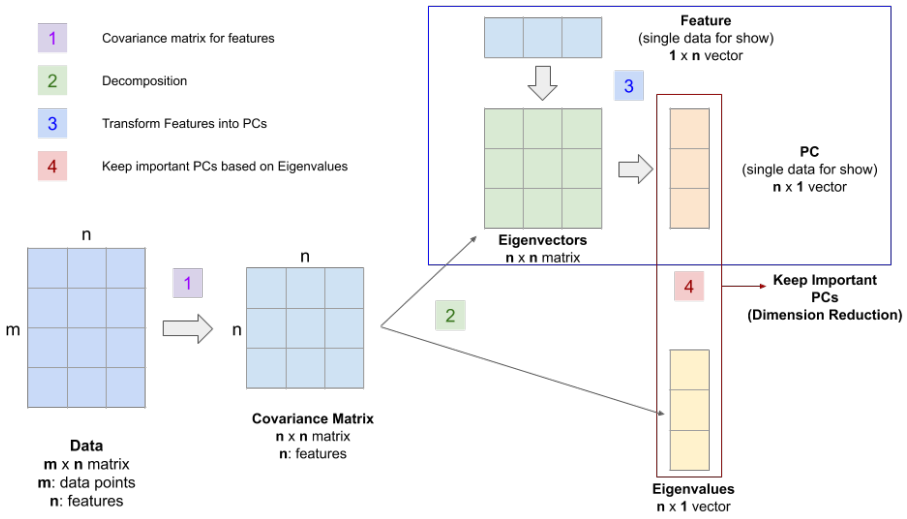


PCA



PCA es una reducción de dimensionalidad que identifica relaciones importantes en nuestros datos, transforma los datos existentes en función de estas relaciones y luego cuantifica la importancia de estas relaciones para que podamos mantener las relaciones más importantes y descartar las demás. Para recordar esta definición, podemos dividirla en cuatro pasos:

1 Identificamos la relación entre las variables predictoras a través de una Matriz de Covarianza.
2 A través de la transformación lineal o autodescomposición de la Matriz de Covarianza, obtenemos autovectores y autovalores.
3 Luego transformamos nuestros datos usando vectores propios en componentes principales.
4 Por último, cuantificamos la importancia de estas relaciones utilizando valores propios y conservamos los componentes principales importantes.



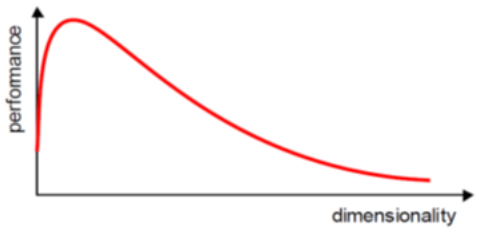
Curse of dimensionality

PCA es extremadamente útil cuando se trabaja con conjuntos de datos que tienen muchas features. Las aplicaciones comunes, como el procesamiento de imágenes, la investigación del genoma siempre tienen que lidiar con miles, si no decenas de miles de columnas.

Si bien tener más datos siempre es excelente, a veces contienen tanta información que tendríamos un tiempo de entrenamiento del modelo increíblemente largo y la "curse of dimensionality" comienza a convertirse en un problema.

Curse of dimensionality

La dimensionalidad en un conjunto de datos se convierte en un impedimento severo para lograr una eficiencia razonable para la mayoría de los algoritmos. Aumentar el número de características no siempre mejora la precisión. Cuando los datos no tienen suficientes características, es probable que el modelo no se ajuste bien, y cuando los datos tienen demasiadas características, es probable que se ajuste demasiado. De ahí que se llame la maldición de la dimensionalidad. La maldición de la dimensionalidad es una paradoja asombrosa para los científicos de datos, basada en la cantidad explosiva de espacios n-dimensionales, a medida que aumenta el número de dimensiones, n.

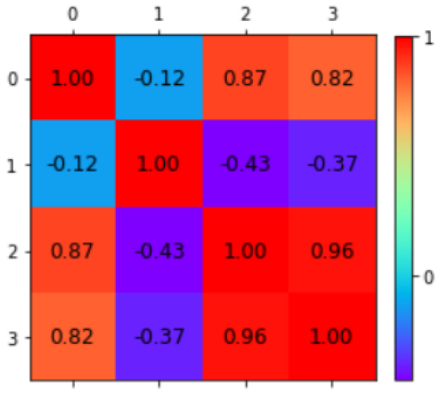


La matriz de covarianza y el mapa de calor:

La matriz de covarianza es el primer paso en la reducción de la dimensionalidad porque da una idea de la cantidad de características que se relacionan fuertemente, y suele ser el primer paso en la reducción de la dimensionalidad porque da una idea de la cantidad de características fuertemente relacionadas para que esas características se puede descartar.

También da el detalle de todas las características independientes. Proporciona una idea de la correlación entre todos los diferentes pares de características.

Matriz de covarianza



Una correlación de la representación del mapa de calor:

1 Entre la primera y la tercera características.
2 Entre la primera y la cuarta características.
3 Entre la tercera y la cuarta característica.

Características independientes:

La segunda característica es casi independiente de las demás.

Aquí, la matriz de correlación y su representación pictórica han dado una idea sobre el número potencial de reducción de características. Por lo tanto, se pueden mantener dos características y se pueden reducir otras características aparte de esas dos características.

Hay dos formas de reducción de la dimensionalidad:

Selección de características:

En la selección de features, por lo general, se selecciona un subconjunto de los features originales.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ . \\ . \\ x_N \end{bmatrix} \rightarrow \mathbf{y} = \begin{bmatrix} x_{i_1} \\ x_{i_2} \\ . \\ . \\ x_{i_K} \end{bmatrix}$$

K<<N

Principal Component Analysis

Extracción de características:

En la extracción de características, se encuentra un conjunto de características nuevas. Eso se encuentra a través de un mapeo de las características existentes. Además, el mapeo puede ser lineal o no lineal.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ . \\ . \\ x_N \end{bmatrix} \xrightarrow{f(\mathbf{x})} \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_K \end{bmatrix}$$

K<<N

¿Cómo funciona PCA?

Podemos comparar PCA con escribir un resumen de un libro.

Encontrar el tiempo para leer un libro de 1000 páginas es un lujo que pocos pueden permitirse. ¿No sería bueno si pudiéramos resumir los puntos más importantes en solo 2 o 3 páginas para que la información sea fácilmente digerible incluso para la persona más ocupada? Es posible que perdamos algo de información en el proceso, pero bueno, al menos tenemos una idea general.

Es un proceso de dos pasos. No podemos escribir un resumen de un libro si no hemos leído o entendido el contenido del libro.

PCA funciona de la misma manera: comprender, luego resumir.

Comprender los datos al estilo PCA

El ser humano entiende el significado de un libro de cuentos mediante el uso del lenguaje expresivo. Desafortunadamente, PCA no habla inglés. Tiene que encontrar

Varianza

significado dentro de nuestros datos a través de su lenguaje preferido, las matemáticas.

¿Puede PCA entender qué parte de nuestros datos es importante?

¿Podemos cuantificar matemáticamente la cantidad de información incrustada en los datos?

Bueno, la variación puede.

Cuanto mayor sea la varianza, mayor será la información y viceversa.

$$S^2 = \frac{\sum(x_i - \bar{x})^2}{n-1}$$

¿Como asociamos la variación con la información? ¿De dónde viene esta asociación? ¿Y por qué debería tener sentido?

Supongamos que estamos jugando un juego de adivinanzas con nuestros amigos. El juego es simple. Nuestros amigos se tapan el rostro y tenemos que adivinar quién es quién basándonos únicamente en su altura. Siendo los buenos amigos que somos, recordamos cuán altos son todos.

Person	Height (cm)
Alex	145
Ben	160
Chris	185

Ahora, intentemos adivinar un grupo diferente de amigos.

Varianza

Person	Height (cm)
Daniel	172
Elsa	173
Fernandez	171

¿Pueden adivinar quién es quién? Es difícil cuando son muy similares en altura.

Anteriormente, no tuvimos problemas para diferenciar una persona de 185 cm de una persona de 160 cm y una persona de 145 cm porque su altura varía mucho.

De la misma manera, cuando nuestros datos tienen una mayor varianza, contienen más información.

PCA se define como una transformación ortogonal lineal que transforma los datos en un nuevo sistema de coordenadas, de modo que la mayor variación por alguna proyección escalar de los datos llega a estar en la primera coordenada (llamado el primer componente principal), la segunda mayor variación en la segunda coordenada, y así sucesivamente.

A los ojos de PCA, la varianza es una forma objetiva y matemática de cuantificar la cantidad de información en nuestros datos.

La varianza es información.

Person	Height (cm)	Weight (kg)
Alex	145	68
Ben	160	67
Chris	185	69

El mismo grupo de amigos y su respectiva altura y peso.

Varianza

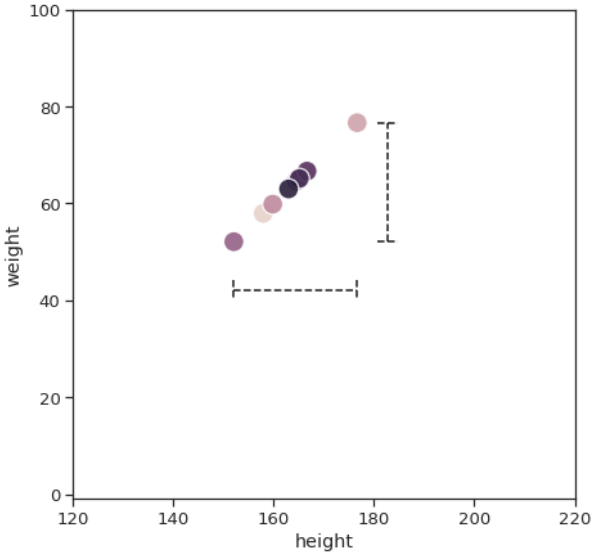
Al principio, solo teníamos altura. Ahora, hemos duplicado la cantidad de datos de nuestros amigos. ¿Cambiaría eso nuestra estrategia?

Resumir datos con PCA:

Las diferencias de peso son tan pequeñas (también conocidas como pequeñas variaciones), que no nos ayudan a diferenciar a nuestros amigos en absoluto. Todavía tenemos que depender principalmente de la altura para hacer nuestras conjeturas.

Intuitivamente, acabamos de reducir nuestros datos de 2 dimensiones a 1 dimensión. La idea es que podamos mantener selectivamente los features con varianzas más altas y luego olvidarnos de las features con varianzas más bajas.

Pero, ¿Qué sucede si la altura y el peso tienen la misma variación? ¿Significa que ya no podemos reducir la dimensionalidad de este conjunto de datos?



La línea punteada representa la variación de altura y peso.

Feature	Variance
Height	1.11
Weight	1.11
TOTAL	**2.22**

Todas las características están estandarizadas a la misma escala para una comparación justa

Componentes principales

En este caso, es muy difícil elegir las variables que queremos eliminar. Si desechamos cualquiera de las variables, estamos desperdiciando la mitad de la información.

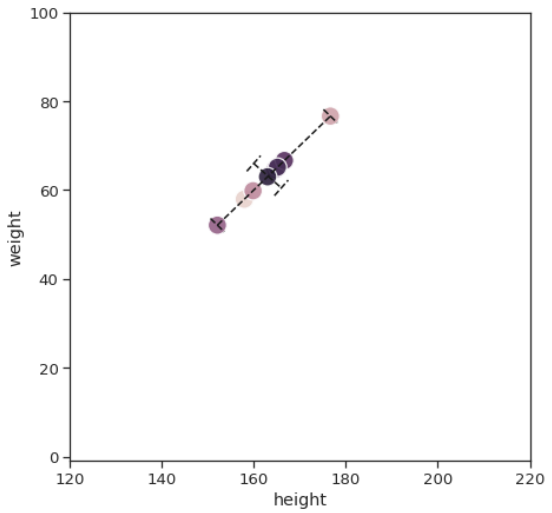
¿Podemos quedarnos con los dos?

Tal vez, con una perspectiva diferente.

Los mejores libros siempre tienen temas ocultos que no están escritos sino implícitos. Leer cada capítulo individualmente no tendría sentido. Pero si lo leemos todo, nos da suficiente contexto para armar los rompecabezas: surge la trama subyacente.

Hasta ahora, solo hemos estado observando la variación de altura y peso individualmente. En lugar de limitarnos a elegir solo uno u otro, ¿por qué no combinarlos?

Cuando miramos más de cerca nuestros datos, la cantidad máxima de variación no se encuentra en el eje x, ni en el eje y, sino en una línea diagonal. La segunda variación más grande sería una línea de 90 grados que atraviesa la primera.

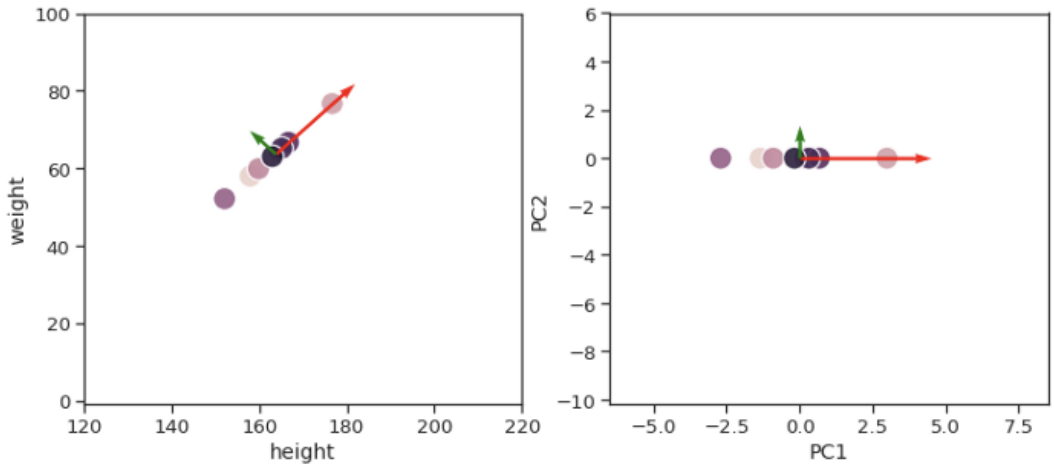


La línea punteada muestra la dirección de la varianza máxima.

Para representar estas 2 líneas, PCA combina la altura y el peso para crear dos variables completamente nuevas. Podría ser 30% de altura y 70% de peso, o 87,2% de altura y 13,8% de peso, o cualquier otra combinación según los datos que tengamos.

Componentes principales

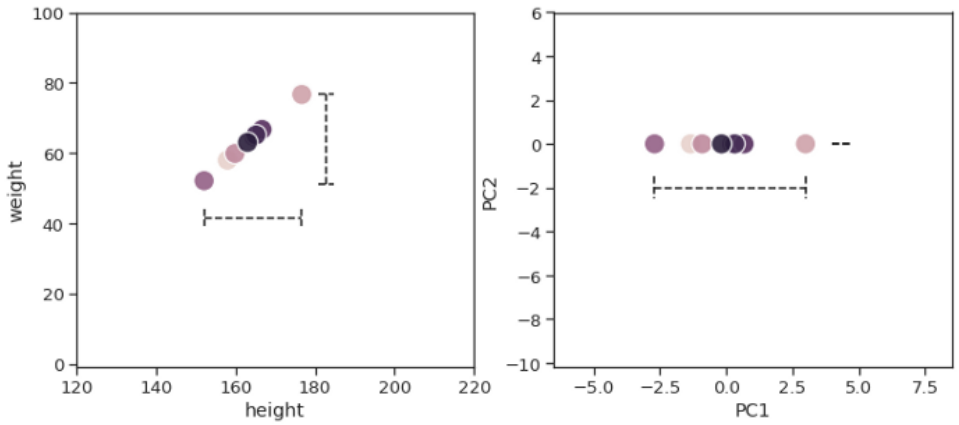
Estas dos nuevas variables se denominan primer componente principal (PC1) y segundo componente principal (PC2). En lugar de usar altura y peso en los dos ejes, podemos usar PC1 y PC2 respectivamente.



(Izquierda) Las flechas roja y verde son los ejes principales en los datos originales.
(Derecha) La dirección de los ejes principales se ha girado para convertirse en los nuevos ejes x e y.

Componentes principales

Echemos un vistazo a las variaciones nuevamente.



(Izquierda) La variación de la altura y el peso son similares en los datos originales.
(Derecha) Después de la transformación PCA, todas las variaciones se muestran en el eje PC1.

Feature	Variance	Feature	Variance
Height	1.11	PC1	2.22
Weight	1.11	PC2	0.00
TOTAL	**2.22**	**TOTAL**	**2.22**

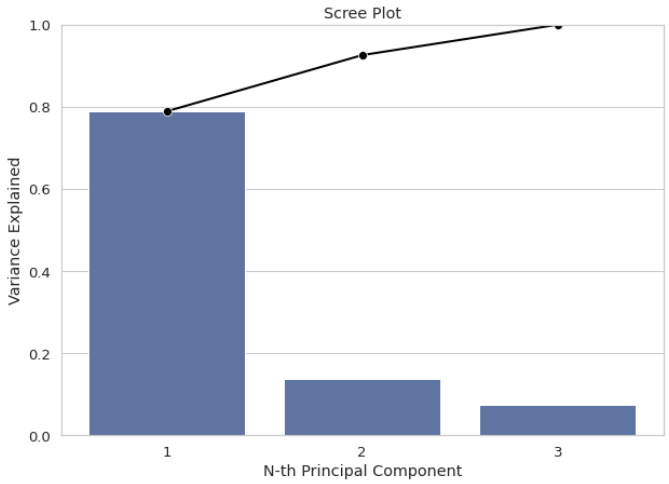
Todas las características están estandarizadas a la misma escala para una comparación justa

PC1 por si sola puede capturar la variación total de altura y peso combinados. Dado que PC1 tiene toda la información, ya sabemos lo que tenemos que hacer, eliminar PC2 y saber que nuestros nuevos datos siguen siendo representativos de los datos originales.

Cuando se trata de datos reales, la mayoría de las veces no obtendremos un componente principal que capture el 100 % de las varianzas. Realizar un PCA nos dará un número N de componentes principales, donde N es igual a la dimensionalidad de nuestros datos originales. De esta lista de componentes principales, generalmente elegimos el menor número de componentes principales que explicarían la mayor parte de nuestros datos originales.

Scree Plot

Una gran ayuda visual que nos ayudará a tomar esta decisión es un Scree Plot.

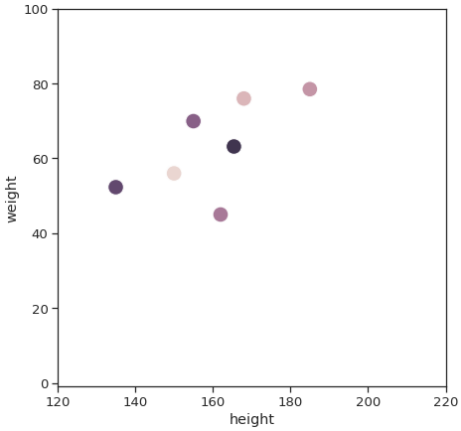


Un ejemplo de un Scree Plot para un conjunto de datos tridimensionales.

El gráfico de barras nos dice la proporción de varianza explicada por cada uno de los componentes principales. Por otro lado, el gráfico de líneas superpuestas nos da la suma acumulada de la varianza explicada hasta el N-ésimo componente principal. Idealmente, queremos obtener al menos un 90% de variación con solo 2 o 3 componentes para que se retenga suficiente información mientras aún podemos visualizar nuestros datos en un gráfico.

Mirando el gráfico estaria bien usar 2 componentes principales.

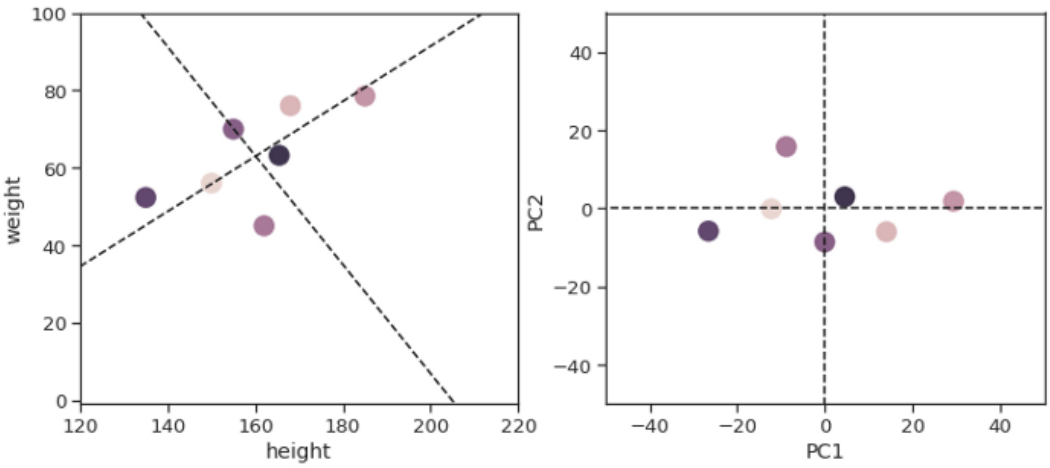
Como no estamos eligiendo todos los componentes principales, inevitablemente perdemos algo de información. Pero no hemos descrito exactamente lo que estamos perdiendo.



Los puntos están dispersos, pero aún podemos ver alguna correlación positiva en una línea diagonal.

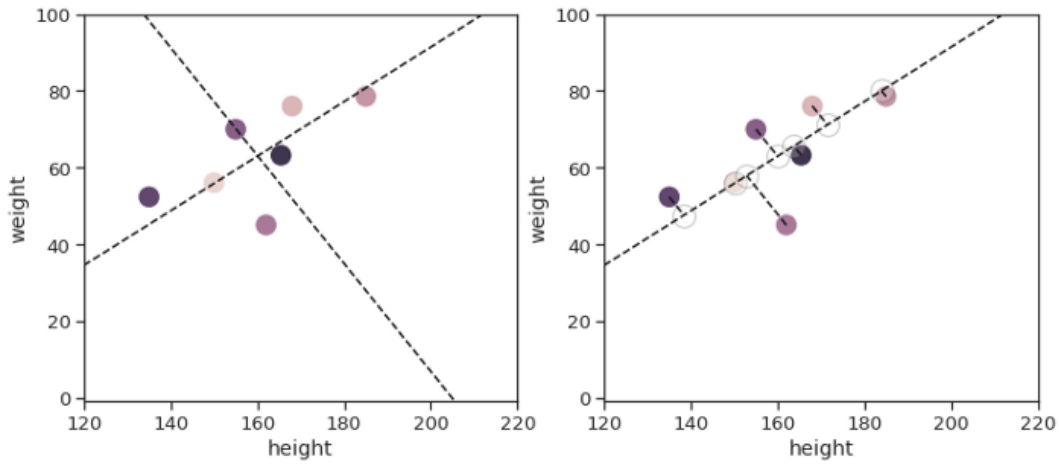
Componentes principales

Si alimentamos nuestros datos a través del modelo PCA, comenzaría dibujando el Primer Componente Principal seguido del Segundo Componente Principal. Cuando transformamos nuestros datos originales de 2 dimensiones a 2 dimensiones, todo permanece igual excepto la orientación. Simplemente rotamos nuestros datos para que la variación máxima esté en PC1.



(Izquierda) Las líneas punteadas son la dirección de los componentes principales primero y segundo.
(Derecha) PCA rota los datos, por lo tanto, coloca la variación máxima en PC1, seguido de PC2.

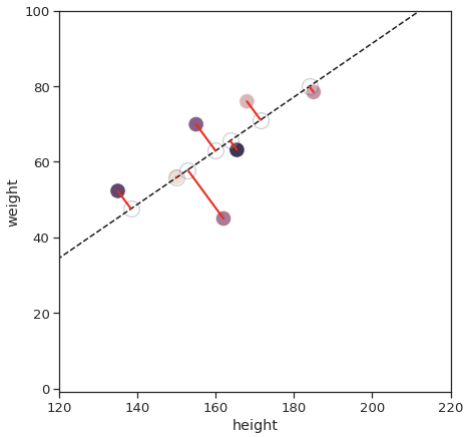
Sin embargo, supongamos que hemos decidido mantener solo el primer componente principal, tendríamos que proyectar todos nuestros puntos de datos en el primer componente principal porque ya no tenemos el eje y.



(Izquierda) Las líneas punteadas son la dirección de los componentes principales primero y segundo.
(Derecha) Todos los puntos ahora se ubican en la línea punteada porque eliminamos el segundo componente principal.

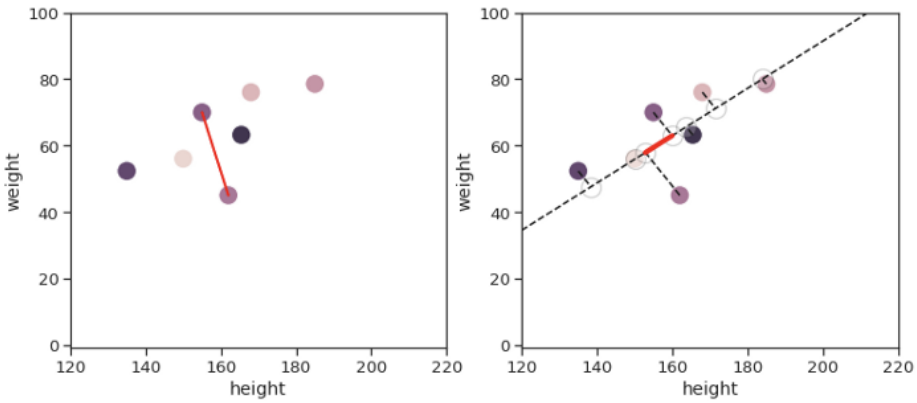
Componentes principales

Lo que perderíamos es la distancia en el Segundo Componente Principal, resaltado con la línea de color rojo debajo.



Todas las líneas rojas son valores en el segundo componente principal y se han eliminado.

Esto tiene implicaciones en la distancia percibida de cada punto de datos. Si observamos la distancia euclidiana entre dos puntos específicos (también conocida como distancia por pares), notará que algunos puntos están mucho más lejos en los datos originales que en los datos transformados.



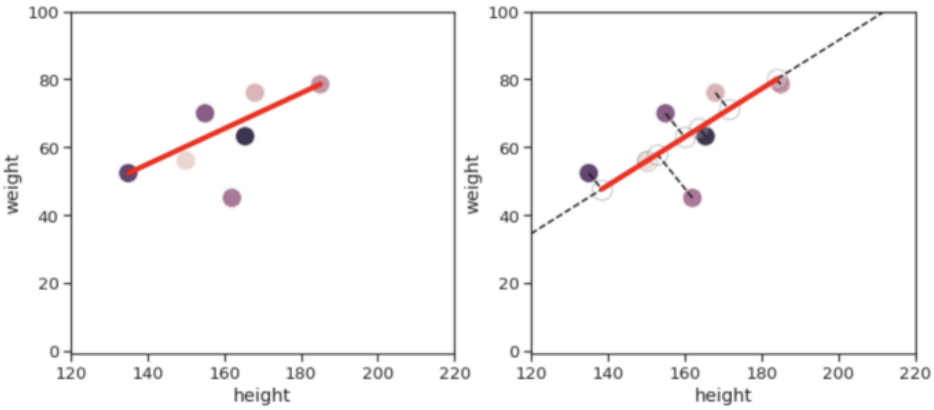
La comparación de la distancia euclidiana por pares entre dos puntos antes (izquierda) y después (derecha) de la dimensionalidad se reduce de 2 a 1 dimensión.

El PCA es una transformación lineal, por lo que en sí mismo no altera las distancias, pero cuando comenzamos a eliminar dimensiones, las distancias se distorsionan.

Se vuelve más complicado: no todas las distancias por pares se ven afectadas por igual.

Componentes principales

Si tomamos los dos puntos más alejados, veremos que son casi paralelos a los ejes principales. Aunque su distancia euclidiana todavía está distorsionada, lo está en un grado mucho menor.



La distancia euclidiana por pares entre estos dos puntos antes (izquierda) y después (derecha) de la reducción de la dimensionalidad sigue siendo bastante idéntica.

La razón es que los ejes de los componentes principales se dibujan en la dirección donde tenemos la varianza más grande. Por definición, la varianza aumenta cuando los puntos de datos están más separados. Entonces, naturalmente, los puntos más alejados se alinearían mejor con los ejes principales.

En resumen, la reducción de dimensiones con PCA cambia las distancias de nuestros datos. Lo hace de una manera que conserva mejor la distancia mayor entre pares que la distancia menor entre pares.

Este es uno de los pocos inconvenientes de reducir dimensiones con PCA y debemos ser conscientes de ello, especialmente cuando se trabaja con el algoritmo basado en la distancia euclidiana.

A veces, puede ser más beneficioso ejecutar nuestro algoritmo en los datos originales, debemos tomar una decisión en función de sus datos y su caso de uso.

ANEXO: ENTROPIA

En termodinámica, la entropía se explica como un estado de incertidumbre o aleatoriedad.

En estadística, tomamos prestado este concepto, ya que se aplica fácilmente al cálculo de probabilidades.

Cuando calculamos la entropía estadística, estamos cuantificando la cantidad de información en un evento, variable o distribución. Comprender esta medida es útil en el aprendizaje automático en muchos casos, como la construcción de árboles de decisión o la elección del mejor modelo clasificador.

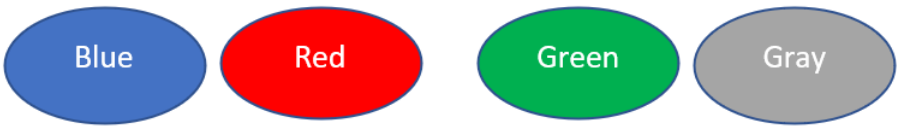
Profundizaremos en la teoría de la entropía y cómo calcularla con el uso de SciPy.

Cálculo de la entropía

El cálculo de la información de una variable fue desarrollado por Claude Shannon, cuyo enfoque responde a la pregunta, ¿cuántas preguntas de "sí" o "no" esperaría hacer para obtener la respuesta correcta?

Considere lanzar una moneda al aire. Suponiendo que la moneda sea justa, tiene 1 posibilidad entre 2 de predecir el resultado. Podrías adivinar cara o cruz, y ya sea que estés en lo correcto o incorrecto, solo necesitas una pregunta para determinar el resultado.

Ahora, digamos que tenemos una bolsa con cuatro discos del mismo tamaño, pero cada uno es de un color diferente:



Para adivinar qué disco se ha extraído de la bolsa, una de las mejores estrategias es eliminar la mitad de los colores. Por ejemplo, comience preguntando si es azul o rojo. Si la respuesta es afirmativa, entonces solo se requiere una pregunta más ya que la respuesta debe ser Azul o Roja. Si la respuesta es no, puede suponer que es verde o gris, por lo que solo se necesita una pregunta más para predecir correctamente el resultado, lo que hace que nuestro total sea de dos preguntas, independientemente de si la respuesta a nuestra pregunta es verde o gris.

ANEXO: ENTROPIA

Podemos ver que cuando es menos probable que ocurra un evento, eligiendo 1 de 4 en lugar de 1 de 2, hay más información para aprender, es decir, se necesitan dos preguntas en lugar de una.

Shannon escribió su cálculo de esta manera:

$$\text{Information}(x) = -\log(p(x))$$

En esta fórmula, log() es un algoritmo de base 2 (porque el resultado es verdadero o falso), y p(x) es la probabilidad de x.

A medida que aumenta el valor de la información, menos predecible se vuelve el resultado.

Cuando una probabilidad es cierta (por ejemplo, una moneda de dos caras que sale cara), la probabilidad es 1.0, lo que produce un cálculo de información de 0.

Probabilidades desiguales

Volviendo al ejemplo de los discos de colores, ¿qué pasa si ahora tenemos 8 discos en la bolsa y no están distribuidos equitativamente?

Color	Quantity
Blue	1
Green	1
Red	2
Gray	4
Total	**8**

Si usamos la estrategia original de eliminar la mitad de los colores preguntando si el disco es azul o verde, nos volvemos menos eficientes ya que hay una probabilidad combinada de 0,25 de que cualquiera de los colores sea correcto en este escenario.

Sabemos que el gris tiene la probabilidad más alta. Usando una estrategia ligeramente diferente, primero preguntamos si Gray es correcto (1 pregunta), luego pasamos a la siguiente probabilidad más alta, Red (2da pregunta), y luego verificamos si es Azul o Verde (3ra pregunta).

ANEXO: ENTROPIA

En este nuevo escenario, sopesar nuestras conjeturas conducirá a que se requiera menos información. Las siguientes tablas muestran la comparación de los dos métodos. La columna de información es el producto de las columnas de Probabilidad y Preguntas.

Equal Guesses			
Color	**Prob**	**Q's**	**Info**
Blue	0.25	2	0.50
Green	0.25	2	0.50
Red	0.25	2	0.50
Gray	0.25	2	0.50
Total	1	8	2.00

Weighted Guesses			
Color	**Prob**	**Q's**	**Info**
Blue	0.125	3	0.375
Green	0.125	3	0.375
Red	0.25	2	0.50
Gray	0.5	1	0.50
Total	1	9	1.75

El método de conjetura igual toma un promedio de 2 preguntas, pero el método de conjetura ponderada toma un promedio de 1.75.

Podemos usar la biblioteca Scipy para realizar el cálculo de entropía. La subbiblioteca de "estadísticas" de Scipy tiene un cálculo de entropía que podemos usar.

ANEXO: ENTROPIA

Otros usos:

El cálculo de entropía se utiliza con éxito en aplicaciones del mundo real en Machine Learning. Aquí hay unos ejemplos.

Árboles de decisión:

Un Árbol de Decisión se basa en un conjunto de decisiones binarias (Verdadero o Falso, Sí o No). Está construido con una serie de nodos donde cada nodo es pregunta: ¿Color == azul? ¿La puntuación de la prueba es > 90? Cada nodo se divide en dos y se descompone en subconjuntos cada vez más pequeños a medida que avanza por el árbol.

La precisión con su árbol de decisión se maximiza al reducir su pérdida. Usar la entropía como función de pérdida es una buena opción aquí. En cada paso que se mueve a través de las ramas, la entropía se calcula antes y después de cada paso. Si la entropía disminuye, el paso se valida. De lo contrario, debe probar en otra rama del arbol.

Clasificación con regresión logística

La clave de una regresión logística es minimizar la pérdida o el error para lograr el mejor ajuste del modelo. La entropía es la función de pérdida estándar para la regresión logística y las redes neuronales.

Uso en series de tiempo:

El concepto de entropía tiene su origen en la física clásica bajo la segunda ley de la termodinámica, una ley que se considera que sustenta nuestra comprensión fundamental del tiempo en la física. Intentar analizar el mundo analógico que nos rodea requiere que midamos el tiempo en pasos discretos, pero hacerlo compromete nuestra capacidad para medir la entropía con precisión. Desde la introducción de la entropía aproximada por Pincus hace tres décadas, el uso de medidas de entropía teóricas de la información para estimar la complejidad, la aleatoriedad o la regularidad de los datos de series temporales se ha vuelto omnipresente en muchos dominios de investigación. Las aplicaciones de la entropía son cada vez mayores, al igual que el número de nuevas entropías que tienen como objetivo estimar la entropía con mayor precisión, menor sensibilidad a la longitud de los datos, fluctuaciones de amplitud, etc.
Para mas detalles revisar el siguiente link:

https://www.entropyhub.xyz/