Trabajo Práctico 2
Reconocimiento de dígitos

Métodos Numéricos

Primer cuatrimestre - 2020

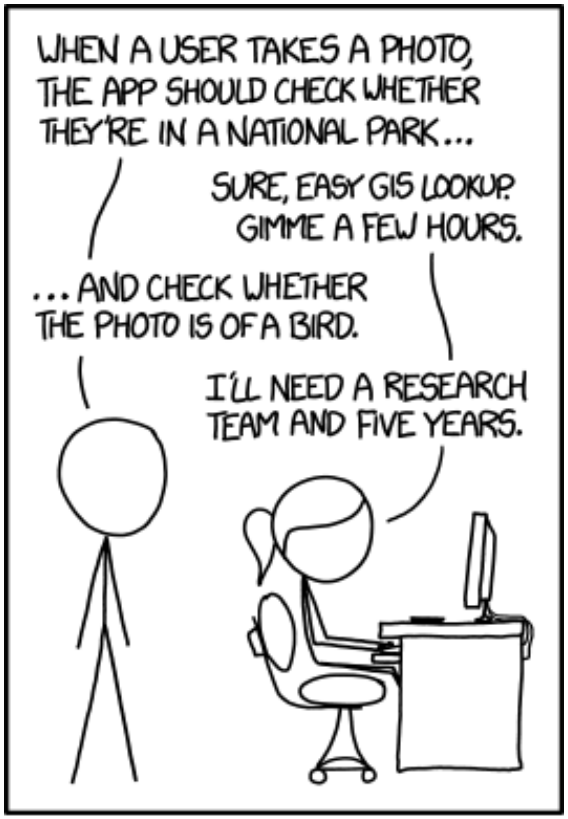
Antes de pasar al TP2...

Donde estamos y qué vimos hasta ahora

- Errores numéricos.
- Resolución de sistema lineales. (EG, LU, SDP)
- Aplicación de resolución de sistemas (Rankings deportivos y otros rankings).
- Cómo experimentar, tanto a nivel metodológico como a nivel implementación.

Subiendonos a la ola: un TP de machine learning

Subiendonos a la ola: un TP de machine learning



Subiendonos a la ola: un TP de machine learning

En realidad ya estabamos en la ola

Reacciones populares

Métodos numéricos

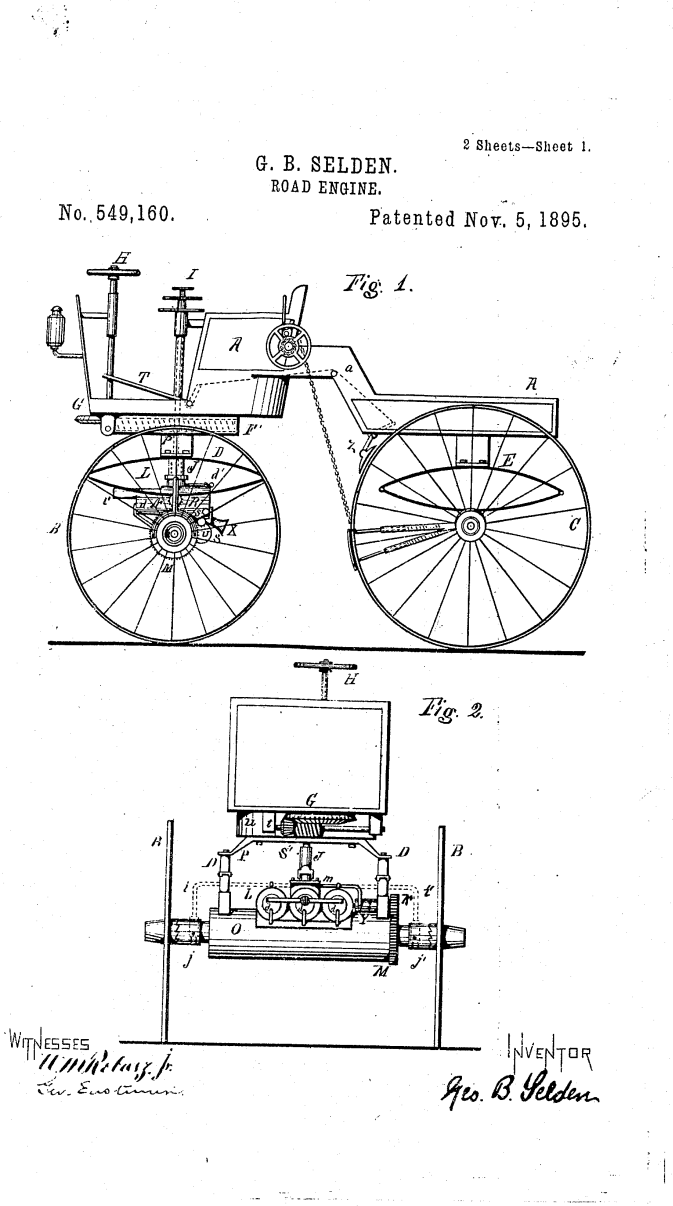
Norma matricial, número de condición, factorización de matrices, distancia de un punto a un subespacio

Machine learning

Data scientist, Big data, Deep learning, Data guru ninja visionary

Trabajo Práctico 2

Reconocimiento de dígitos - Aplicaciones



2 Sheets—Sheet 1.

G. B. SELDEN.

ROAD ENGINE.

No. 549,160. Patented Nov. 5, 1895.

Fig. 1.

Fig. 2.

WITNESSES

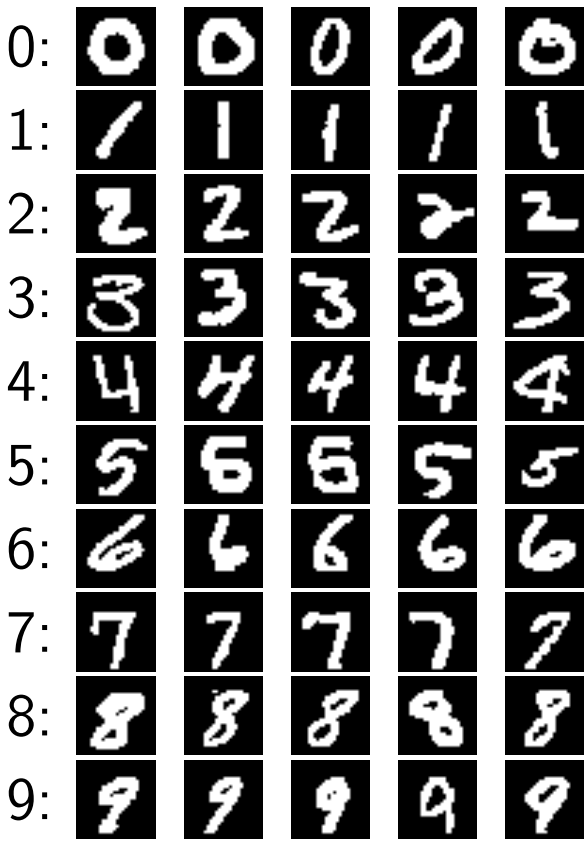
INVENTOR

Geo. B. Selden

Trabajo Práctico 2

Reconocimiento de dígitos

- Datos: base de datos etiquetada de imágenes de dígitos manuscritos (0-9) tomadas de una forma particular.
- Objetivo: dada una nueva imagen de un dígito, ¿A cuál corresponde?



Problema a resolver

Recibimos un nuevo dígito manuscrito, ¿Podemos determinar automáticamente a cuál pertenece?

Reconocimiento de dígitos

Contexto

Objetivo

Desarrollar (no solo en términos de implementación) un *clasificador* que permita reconocer dígitos manuscritos.

Contexto

- Disponemos de una base de datos etiquetada (train), y un conjunto de datos para los que no conocemos cuál es su etiqueta (test). Este último nos permitirá evaluar como se comporta nuestro clasificador.
- Consideramos la base MNIST, en la versión utilizada en *Kaggle*. 42k dígitos en train, 18k dígitos en test.
- Cada dígito es una imagen en escala de grises de 28×28.

Reconocimiento de dígitos

Vecino más cercano

Idea general (caso particular reconocimiento dígitos)

- Consideramos cada imagen como un vector $x_i \in \mathbb{R}^m$, $m = 28 \times 28$, $i = 1, \ldots, n$. Para las imágenes en la base de datos, sabemos además a que clase pertenece.
- Cuando llega una nueva imagen de un dígito z, con el mismo formato, recorremos toda la base y buscamos aquella que minimice

$$\arg\min_{i=1,\ldots,n} \|z - x_i\|_2$$

Luego, le asignamos la clase del representante seleccionado.

Generalización

Considerar más de un vecino.

Reconocimiento de dígitos

Vecinos más cercanos: *kNN*

- Consideramos los k vecinos más cercanos.
- Entre ellos hacemos una votación, eligiendo como clase la *moda* del conjunto. En otras palabras, hacemos una votación y se elige aquella clase con más votos.

Reconocimiento de dígitos

kNN: Ejemplo de clasificación y definición de fronteras

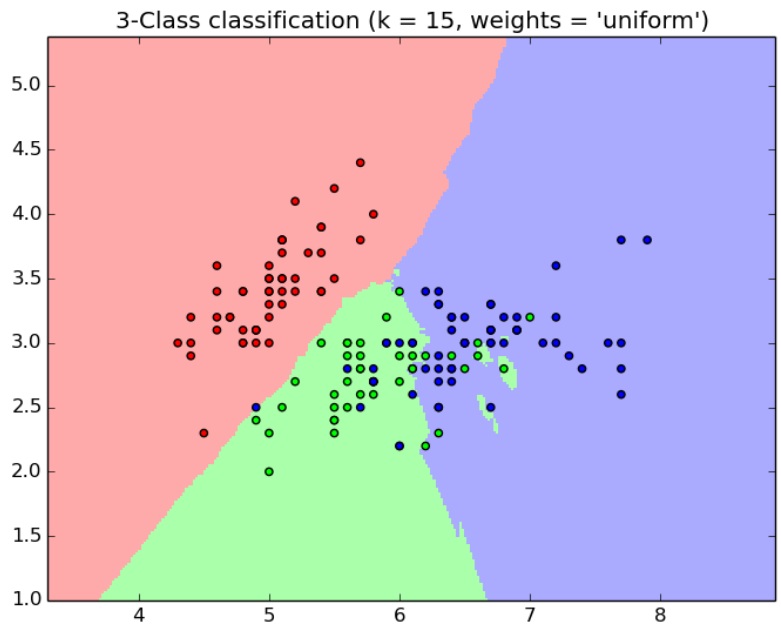


Imagen tomada de SCIKIT-LEARN.ORG

Algunos pros & cons

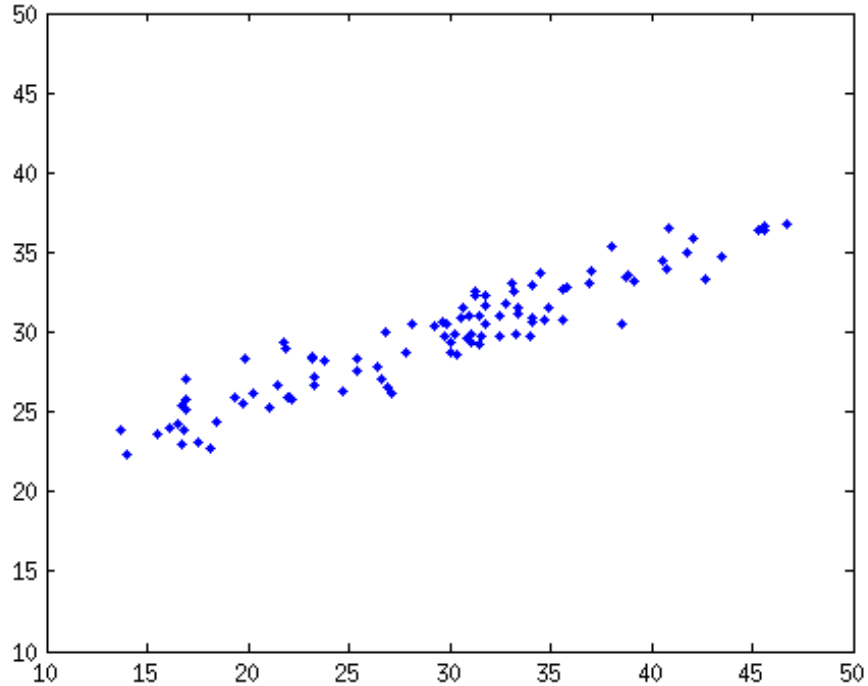
+ Es conceptualmente simple.
+ Funciona bien en general para dimensiones bajas, y puede ser utilizado con pocos ejemplos.
- Sufre de *La maldición de la dimensionalidad.*
- La clasificación puede ser lenta dependiendo del contexto.

Análisis de Componentes Principales

Ejemplo datos en $\mathbb{R}^2$

Sean $x^{(1)}, x^{(2)}, \ldots, x^{(n)}$ una secuencia de n datos, con $x^{(i)} \in \mathbb{R}^2$.

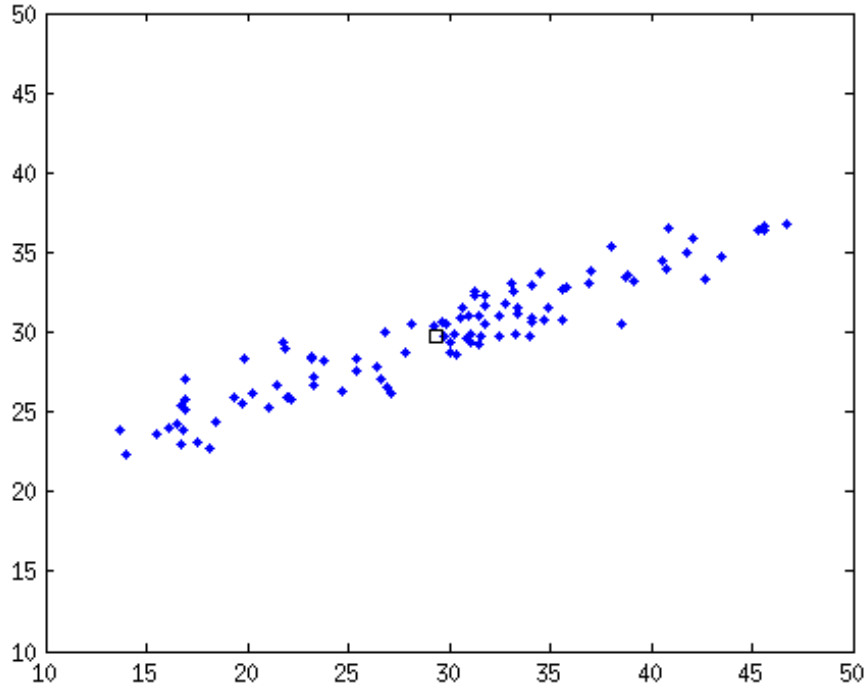
$$X = \begin{bmatrix} {x^{(1)}}^t \\ {x^{(2)}}^t \\ {x^{(3)}}^t \\ {x^{(4)}}^t \\ {x^{(5)}}^t \\ {x^{(6)}}^t \\ \vdots \\ {x^{(n)}}^t \end{bmatrix} = \begin{bmatrix} 26.4320 & 27.7740 \\ 26.8846 & 26.5631 \\ 23.3309 & 26.6983 \\ 30.6387 & 31.5619 \\ 30.5171 & 30.8993 \\ 45.6364 & 36.6035 \\ \vdots & \vdots \\ 16.0650 & 24.0210 \end{bmatrix}$$



Análisis de Componentes Principales

Ejemplo datos en $\mathbb{R}^2$

$$X = \begin{bmatrix} 26.4320 & 27.7740 \\ 26.8846 & 26.5631 \\ 23.3309 & 26.6983 \\ 30.6387 & 31.5619 \\ 30.5171 & 30.8993 \\ 45.6364 & 36.6035 \\ \vdots & \vdots \\ 16.0650 & 24.0210 \end{bmatrix}$$



Media:

$\mu = \frac{1}{n}(x^{(1)} + \cdots + x^{(n)})$

$\mu = (29.3623, 29.7148)$

Varianza de una variable x_k: Medida para la dispersión de los datos.

$\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_k^{(i)} - \mu_k)^2$

$\sigma_{x_1}^2 = 66.2134,\ \sigma_{x_2}^2 = 12.5491$

Análisis de Componentes Principales

Ejemplo datos en $\mathbb{R}^2$ - Covarianza

$$X = \begin{bmatrix} 26.4320 & 27.7740 \\ 26.8846 & 26.5631 \\ 23.3309 & 26.6983 \\ 30.6387 & 31.5619 \\ 30.5171 & 30.8993 \\ 45.6364 & 36.6035 \\ \vdots & \vdots \\ 16.0650 & 24.0210 \end{bmatrix}$$

Covarianza: Medida de cuánto dos variables varían de forma similar. Variables con mayor covarianza inducen la presencia de cierta dependencia o relación.

$$\sigma_{x_j x_k} = \frac{1}{n-1} \sum_{i=1}^{n} (x_j^{(i)} - \mu_j)(x_k^{(i)} - \mu_k)$$

Análisis de Componentes Principales

Ejemplo datos en $\mathbb{R}^2$ - Covarianza

Dadas n observaciones de dos variables x_k, x_j, y $v = (1, \ldots, 1)^t$:

$$\sigma_{x_j x_k} = \frac{1}{n-1} \sum_{i=1}^{n} (x_j^{(i)} - \mu_j)(x_k^{(i)} - \mu_k) = \frac{1}{n-1} (x_k - \mu_k v)^t (x_j - \mu_j v)$$

Matriz de Covarianza:

$$X = \begin{bmatrix} 26.4320 - \mu_1 & 27.7740 - \mu_2 \\ 26.8846 - \mu_1 & 26.5631 - \mu_2 \\ 23.3309 - \mu_1 & 26.6983 - \mu_2 \\ 30.6387 - \mu_1 & 31.5619 - \mu_2 \\ 30.5171 - \mu_1 & 30.8993 - \mu_2 \\ 45.6364 - \mu_1 & 36.6035 - \mu_2 \\ \vdots & \vdots \\ 16.0650 - \mu_1 & 24.0210 - \mu_2 \end{bmatrix}$$

$$M_X = \frac{1}{n-1} X^t X = \begin{bmatrix} \sigma_{x_1 x_1} & \sigma_{x_1 x_2} \\ \sigma_{x_1 x_2} & \sigma_{x_2 x_2} \end{bmatrix}$$

$$= \begin{bmatrix} \sigma_{x_1}^2 & \sigma_{x_1 x_2} \\ \sigma_{x_1 x_2} & \sigma_{x_2}^2 \end{bmatrix}$$

$$M_X = \begin{bmatrix} 66.2134 & 27.1263 \\ 27.1263 & 12.5491 \end{bmatrix}$$

¿Cómo expresar mejor nuestros datos?

Objetivo

Buscamos una transformación de los datos que disminuya la redundancia (es decir, disminuir la covarianza).

- Cambio de base: $\hat{X}^t = PX^t$.
- Cómo podemos hacerlo? Diagonalizar la matriz de covarianza. Esta matriz tiene la varianza de cada variable en la diagonal, y la covarianza en las restantes posiciones. Luego, al diagonalizar buscamos variables que tengan covarianza cero entre sí y la mayor varianza posible.

Autovalores y Autovectores

Definición

Sea $A \in \mathbb{R}^{n \times n}$. Un *autovector* de A es un vector no nulo tal que $Ax = \lambda x$, para algun escalar λ. Un escalar λ es denominado *autovalor* de A si existe una solución no trivial x del sistema $Ax = \lambda x$. En este caso, x es llamado *autovector asociado a* λ.

Consideramos:

$$A = \begin{bmatrix} 3 & -2 \\ 1 & 0 \end{bmatrix}, u = \begin{bmatrix} -1 \\ 1 \end{bmatrix}, v = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

$$Au = \begin{bmatrix} -5 \\ -1 \end{bmatrix}, Av = \begin{bmatrix} 4 \\ 2 \end{bmatrix} = 2 \begin{bmatrix} 2 \\ 1 \end{bmatrix} = 2v$$

Gráficamente....A sólo estira (o encoge) el vector v.

Diagonalización

En muchos casos, la presencia de autovectores-autovalores puede ser utilizada para encontrar una factorización $A = PDP^{-1}$, donde D es una matriz diagonal.

Intuición

Podemos encontrar una base donde la transformación lineal A se comporta como si fuese diagonal.

Observación

No toda matriz $A \in \mathbb{R}^{n \times n}$ es diagonalizable.

Teorema

Una matriz $A \in \mathbb{R}^{n \times n}$ es diagonalizable sí y solo sí A tiene n autovectores linealmente independientes (las columnas de P).

Teorema

Si $A \in \mathbb{R}^{n \times n}$ es simétrica, entonces existe una base ortonormal de autovectores $\{v_1, \ldots, v_n\}$ asociados a $\lambda_1, \ldots, \lambda_n$.

Consecuencia: Existe P, y $P^{-1} = P^t$. Luego, $A = PDP^t$.

Cálculo de autovalores/autovectores

- Vamos a necesitar calcular los autovectores v de una matriz para poder calcular las transformaciónes de los métodos que estamos viendo.
- Consideremos A^tA, y supongamos $\lambda_1 > \lambda_2 > \cdots > \lambda_k$. A^tA es simétrica y semidefinida positiva.
- Podemos considerar el Método de la Potencia para calcular λ_1 y v_1.

1. MetodoPotencia(B,x_0,niter)
2. $v \leftarrow x_0$.
3. Para $i = 1, \ldots, niter$
4. $v \leftarrow \frac{Bv}{||Bv||}$
5. Fin Para
6. $\lambda \leftarrow \frac{v^tBv}{v^tv}$
7. Devolver λ, v.

Cálculo de autovalores/autovectores

Una vez que tenemos λ_1 y v_1, como seguimos?

Deflación

Sea $B \in \mathbb{R}^{n \times n}$ una matriz con autovalores distintos $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|$ y una base ortonormal de autovectores. Entonces, la matriz $B - \lambda_1 v_1 v_1^t$ tiene autovalores $0, \lambda_2, \ldots, \lambda_n$ con autovectores asociados $v_1, \ldots, v_n$.

- $(B - \lambda_1 v_1 v_1^t) v_1 = B v_1 - \lambda_1 v_1 (v_1^t v_1) = \lambda_1 v_1 - \lambda_1 v_1 = 0 v_1$.
- $(B - \lambda_1 v_1 v_1^t) v_i = B v_i - \lambda_1 v_1 (v_1^t v_i) = \lambda_i v_i$.

Observación

En nuestro caso, no hace falta que todos los autovalores tengan magnitudes distintas.

¿Cómo expresar mejor nuestros datos?

- Cambio de base: $\hat{X}^t = PX^t$.

  Sea P ortogonal y $M_{\hat{X}}$ la matriz de covarianza de $\hat{X}$.

$$\begin{aligned} M_{\hat{X}} &= \frac{1}{n-1}\hat{X}^t\hat{X} \\ &= \frac{1}{n-1}(PX^t)(XP^t) \\ &= P\frac{X^tX}{n-1}P^t \\ &= PM_XP^t \end{aligned}$$

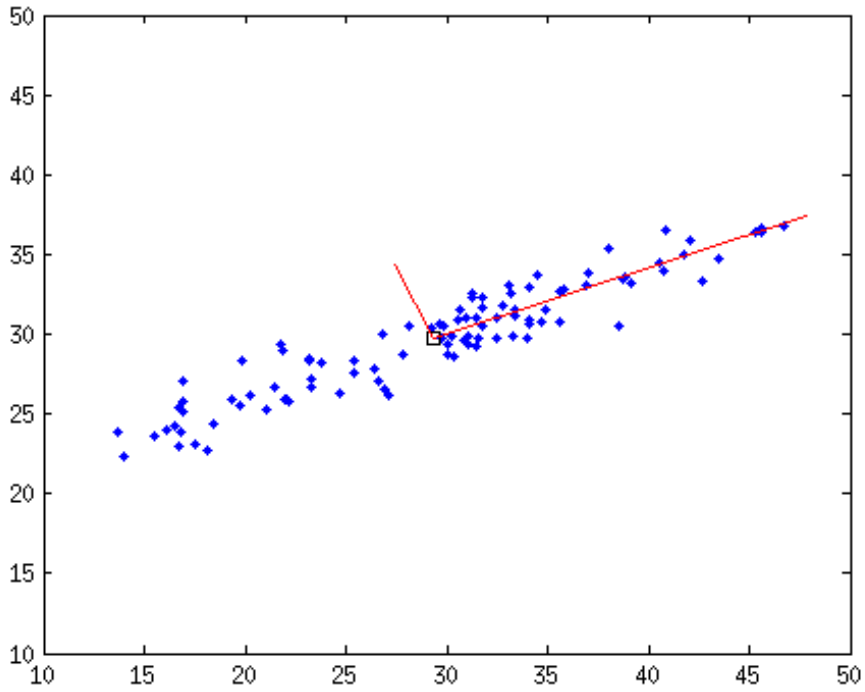
- M_X es simétrica, entonces existe V ortogonal tal que $M_X = VDV^t$.

$$\begin{aligned} M_{\hat{X}} &= PM_XP^t \\ &= P(VDV^t)P^t \qquad \text{tomamos } P = V^t \\ &= (V^tV)D(VV^t) = D \end{aligned}$$

¿Cómo expresar mejor nuestros datos?

Volvemos al ejemplo

$$
\begin{aligned}
M_X &= \begin{bmatrix} 66.2134 & 27.1263 \\ 27.1263 & 12.5491 \end{bmatrix} \\
&= \underbrace{\begin{bmatrix} 0.9228 & -0.3852 \\ 0.3852 & 0.9228 \end{bmatrix}}_{V} \underbrace{\begin{bmatrix} 77.5362 & 0 \\ 0 & 1.2263 \end{bmatrix}}_{D=M_{\hat{X}}} \underbrace{\begin{bmatrix} 0.9228 & 0.3852 \\ -0.3852 & 0.9228 \end{bmatrix}}_{V^t}
\end{aligned}
$$



Análisis de Componentes Principales

Resumen hasta acá

- Tenemos n muestras de m variables.
- Calculamos el vector μ que contiene la media de cada de una las variables.
- Construimos la matriz $X \in \mathbb{R}^{n \times m}$ donde cada muestra corresponde a una fila de X y tienen media cero (i.e., $x^{(i)} := (x^{(i)} - \mu)/\sqrt{n-1}$).
- Diagonalizamos la matriz de covarianzas M_X. La matriz V (ortogonal) contiene los autovectores de M_X.

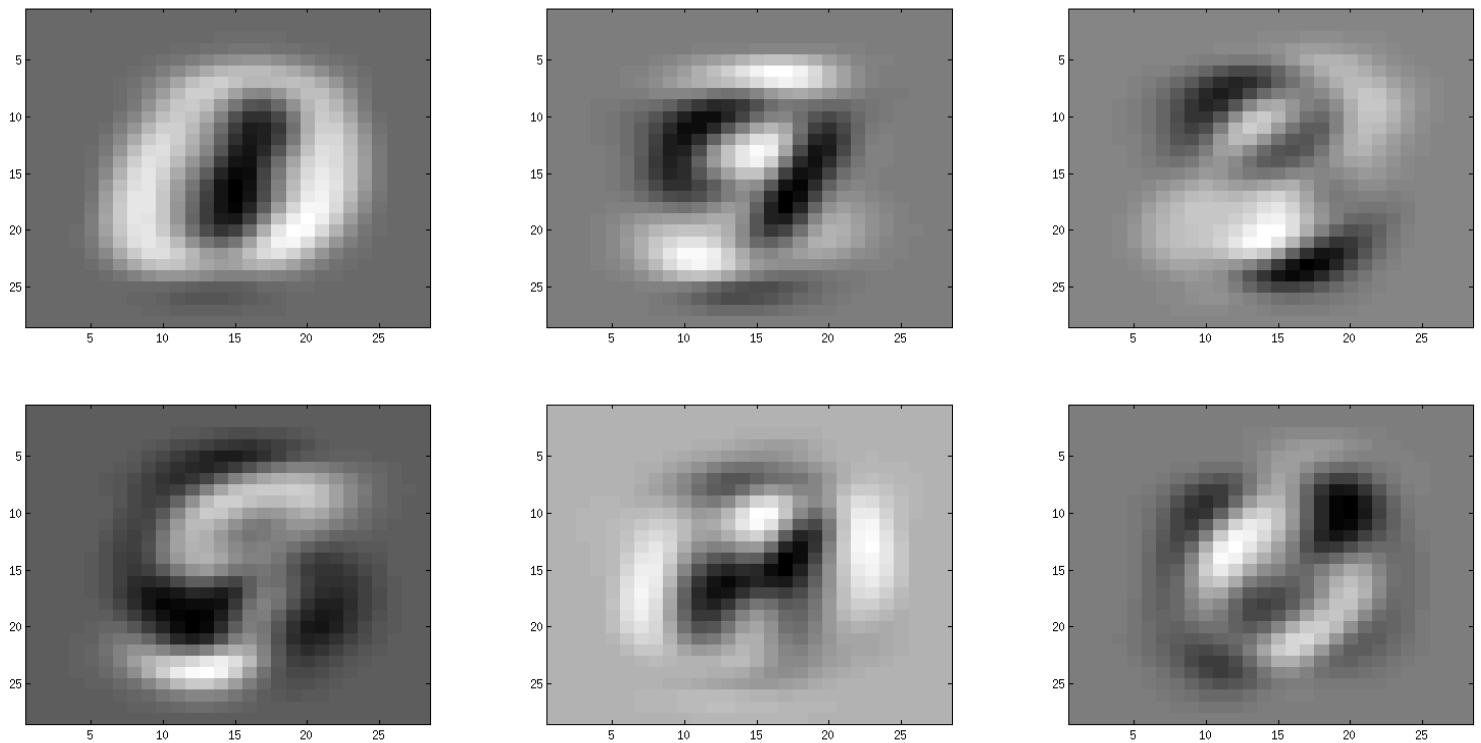
Propiedades del cambio de base

- Disminuye redundancias.
- El cambio de base $\hat{X}^t = PX^t = V^t X^t$ asigna a cada muestra un nuevo *nombre* mediante un cambio de coordenadas.
- Las columnas de V (autovectores de M_X) son las componentes principales de los datos.
- En caso de m grande, es posible tomar sólo un subconjunto de las componentes principales para estudiar (i.e., aquellas que capturen mayor proporción de la varianza de los datos).

Reconocimiento de dígitos

Autodígitos (Eigendigits)

Los primeros 6 autovectores en V.



Reconocimiento de dígitos

¿Cómo reconocemos un dígito?

Idea

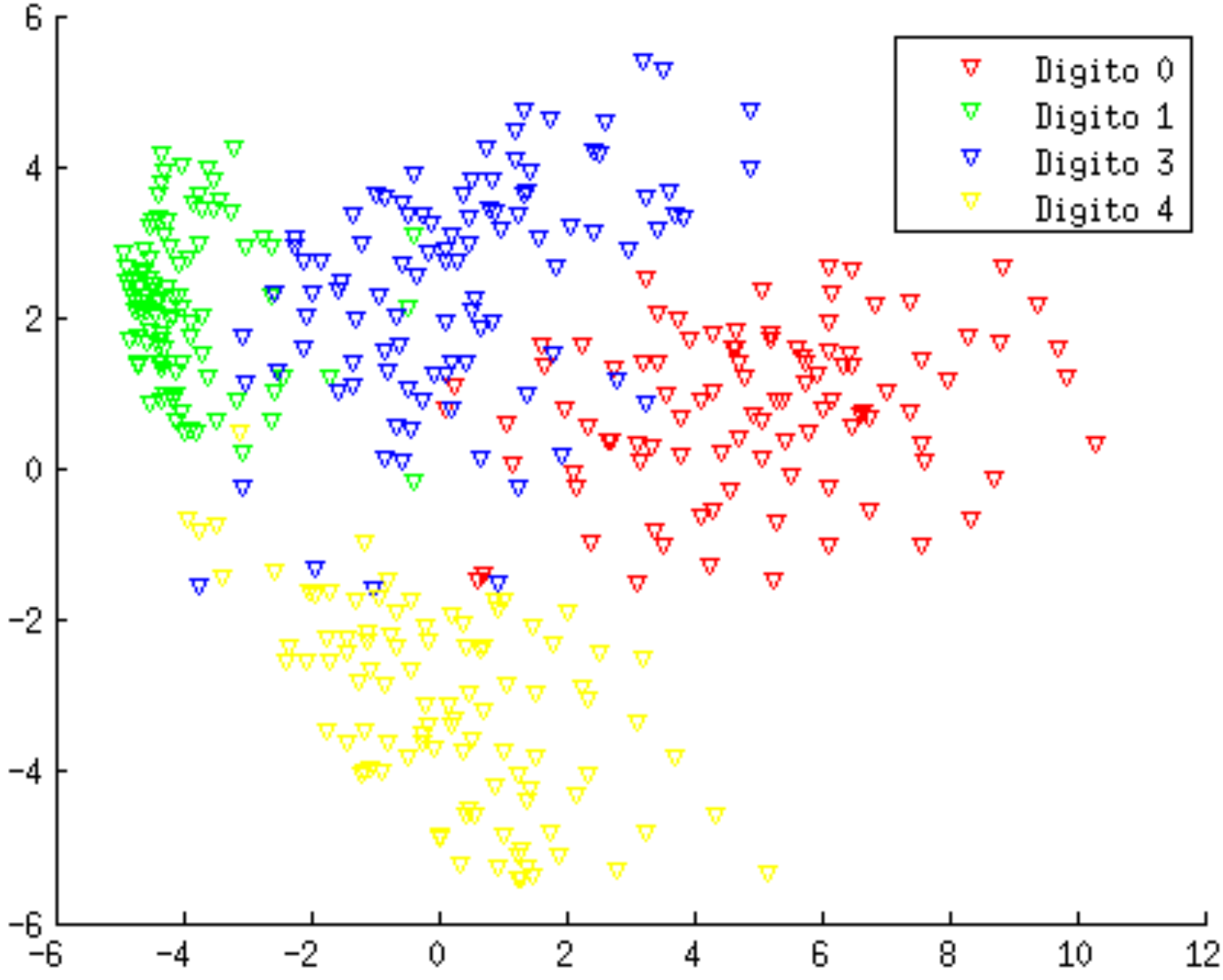
- Utilizar el cambio de base, transformando cada imagen convenientemente.
- Reducir la dimensión de los datos utilizando sólo algunas de las nuevas variables (eligiendo aquellas que capturan una fracción mayor de la varianza).

Procedimiento

- Reducción de la dimensión: parámetro de entrada que indica cuántas componentes principales considerar, α. Es decir, tomaremos $\bar{V} = [v_1 \ v_2 \ \dots \ v_\alpha]$.
- Tranformación característica: Aplicamos el cambio de base a cada muestra $x^{(i)}$, definimos $tc(x^{(i)}) = \bar{V}^t x^{(i)} = (v_1^t x^{(i)}, \dots, v_\alpha^t x^{(i)})$.

Reconocimiento de dígitos

Transformación + Reducción ($k = 2$)



Reconocimiento de dígitos

¿Cómo reconocemos un dígito?

Finalmente, dada una imagen de un dígito que no se encuentra en la base:

- Vectorizamos la imagen en $x^* \in \mathbb{R}^m$.
- Definimos $\bar{x}^* = (x^* - \mu)/\sqrt{n-1}$.
- Aplicamos la transformación característica, $tc(\bar{x}^*)$ y buscamos (de alguna manera) a que dígito pertenece.

Pregunta:

Sugerencias para buscar a qué dígito pertenece?

Reconocimiento de dígitos

Metodología de evaluación

Elegimos un numero de vecinos k (adicionalmente un número α o γ de componentes). Como evaluamos si el método funciona?

- Como medimos la efectividad del método?

Reconocimiento de dígitos

Metodología de evaluación

Elegimos un numero de vecinos k (adicionalmente un número α o γ de componentes). Como evaluamos si el método funciona?

- Como medimos la efectividad del método?
- Tiene sentido probarlo sobre la base de training?

Reconocimiento de dígitos

Metodología de evaluación

Elegimos un numero de vecinos k (adicionalmente un número α o γ de componentes). Como evaluamos si el método funciona?

- Como medimos la efectividad del método?
- Tiene sentido probarlo sobre la base de training?
- De alguna forma defino una instancia, pruebo todas las combinaciones de parámetros sobre la misma. Es correcto? Puede surgir algún problema?

Reconocimiento de dígitos

Metodología de evaluación

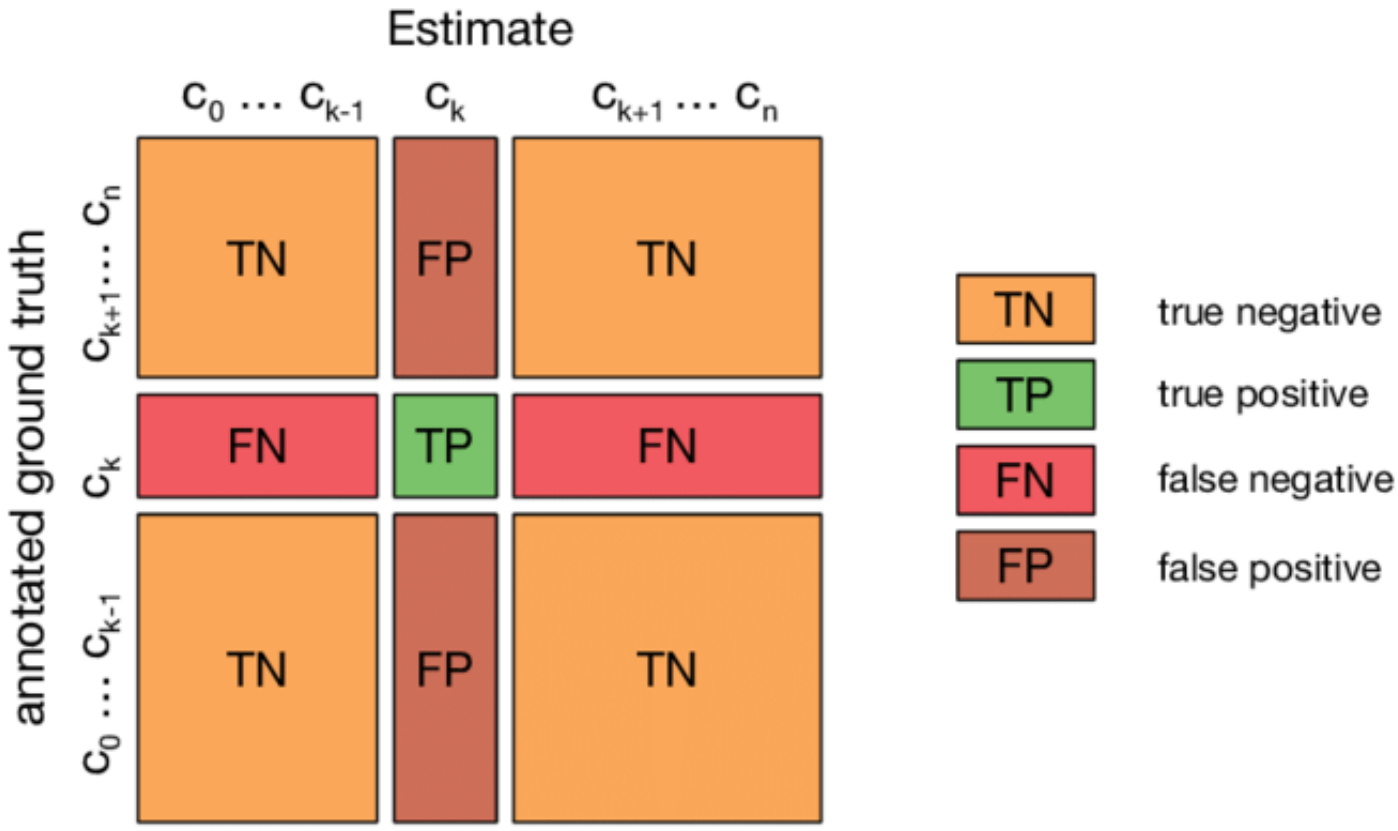
Elegimos un numero de vecinos k (adicionalmente un número α o γ de componentes). Como evaluamos si el método funciona?

- Como medimos la efectividad del método?
- Tiene sentido probarlo sobre la base de training?
- De alguna forma defino una instancia, pruebo todas las combinaciones de parámetros sobre la misma. Es correcto? Puede surgir algún problema?

Idea

Utilizar la base de entrenamiento convenientemente para estimar y proveer suficiente evidencia respecto a la efectividad del método.

Midiendo la efectividad - Matriz de confusión



Métricas

- **Accuracy**: Los aciertos totales sobre los casos totales. En términos de la matriz de confusión, sumar la diagonal dividido la suma de todas las celdas.
- **Precision**: Aciertos relativos dentro de una clase. Dada una clase i, $\frac{tp_i}{tp_i+fp_i}$.
  La *precision* en el caso de un clasificador de muchas clases, se define como el promedio de las *precision* para cada una de las clases.
- **Recall**: Métrica para medir los reconocimientos dentro de una clase. Dada una clase i, $\frac{tp_i}{tp_i+fn_i}$.

Métricas

- **F1-Score**: Dado que *precision* y *recall* son dos medidas importantes que no necesariamente tienen la misma calidad para un mismo clasificador, se define la métrica F1 para medir un compromiso entre el *recall* y la *precision*. La métrica $F1$ se define como $2 * precision * recall/(precision + recall)$.
- **Kappa de Cohen**: Es una medida para indicar cuánto concuerdan dos clasificadores sobre un mismo set de datos. Dicha medida se define como $\kappa = (p_o - p_a)/(1 - p_a)$. Donde p_o es la probabilidad observada de que los dos clasificadores concuerden y p_a es la probabilidad aleatoria de que lo hagan. Esta métrica puede utilizarse para determinar si el problema contiene ejemplos particularmente complicados, porque por ejemplo ningún clasificador lo reconoce correctamente.

¿Qué hay que hacer en el TP?

Objetivos generales

- Implementar el método *kNN*.
- Implementar el método de PCA y combinarlo con *kNN*.
- Experimentar variando: k, α, K, Analizar los resultados en términos de diferentes métricas (mirando al menos la tasa de efectividad) aplicando *cross validation* sobre la base de training.
- Para encontrar los autovectores necesarios, utilizar el *Método de la Potencia* + *Deflación*.

¿Qué hay que hacer en el TP?

Objetivos generales

- Implementar el método *kNN*.
- Implementar el método de PCA y combinarlo con *kNN*.
- Experimentar variando: k, α, K, Analizar los resultados en términos de diferentes métricas (mirando al menos la tasa de efectividad) aplicando *cross validation* sobre la base de training.
- Para encontrar los autovectores necesarios, utilizar el *Método de la Potencia* + *Deflación*.

Algunas (posibles) preguntas y dificultades

- *kNN* y 42k imágenes de 28×28?
- Tolerancia de corte Método de la Potencia? Se cumplen las condiciones para aplicar deflación?
- Cuántas componentes principales tomar?
- Que combinación de parámetros (modelo) da los mejores resultados?

Implementación

- Eigen: Librería en C++ para hacer cálculos de álgebra lineal. (https://eigen.tuxfamily.org/dox/GettingStarted.html)

How to "install" Eigen?

In order to use **Eigen**, you just need to download and extract **Eigen**'s source code (see the wiki for download instructions). In fact, the header files in the **Eigen** subdirectory are the only files required to compile programs using **Eigen**. The header files are the same for all platforms. It is not necessary to use CMake or install anything.

A simple first program

Here is a rather simple program to get you started.

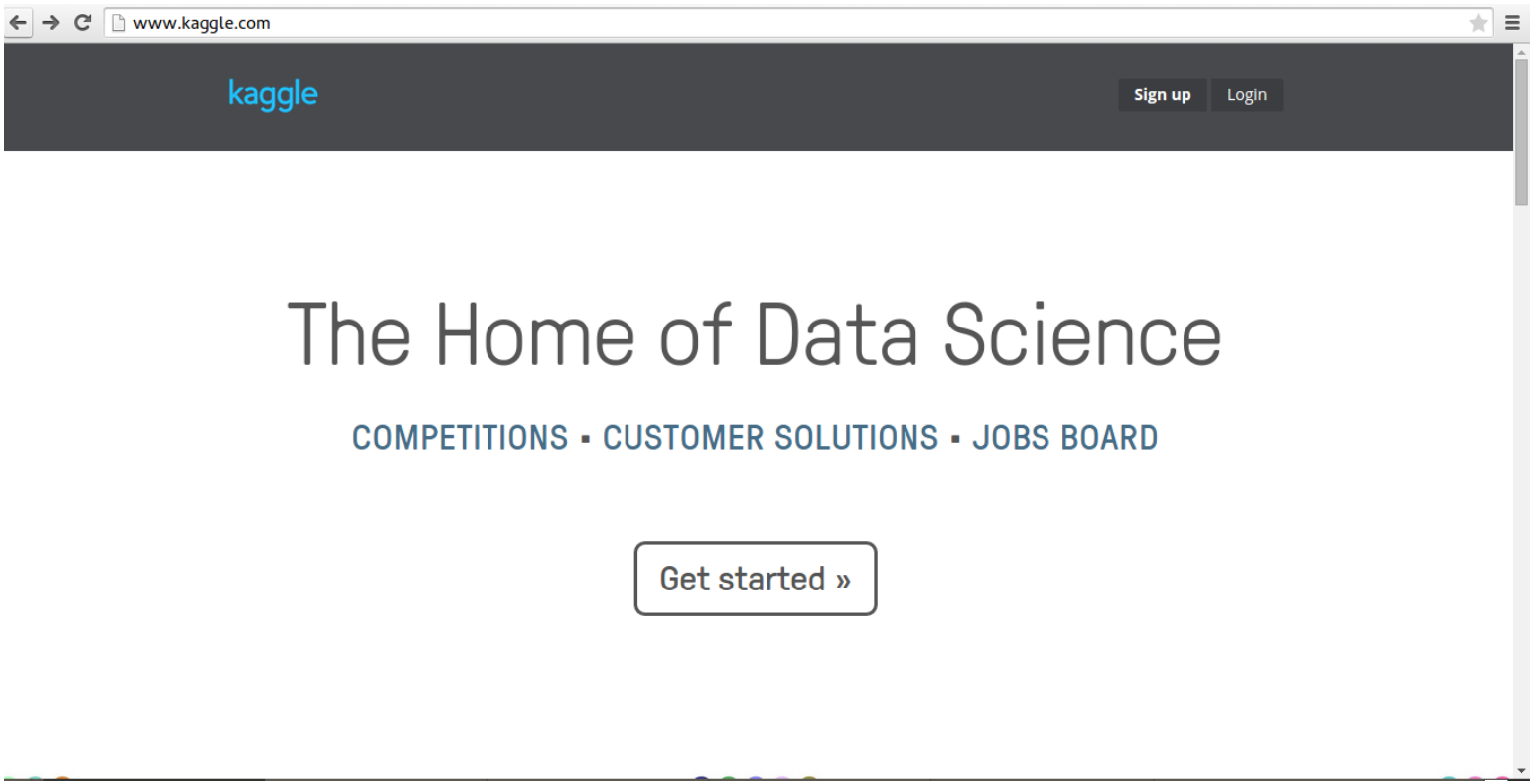
```
#include <iostream>
#include <Eigen/Dense>

using Eigen::MatrixXd;

int main()
{
  MatrixXd m(2,2);
  m(0,0) = 3;
  m(1,0) = 2.5;
  m(0,1) = -1;
  m(1,1) = m(1,0) + m(0,1);
  std::cout << m << std::endl;
}
```

Implementación

- Pybind: Librería para *bindear* código en C++ y Python.
- Esqueleto del tp:

Por último...

Competencia activa en KAGGLE.COM



Entrega

Fecha de entrega

- Formato electrónico: Jueves 11 de Junio de 2020, hasta las 23:59 hs., enviando el trabajo (informe+código) a `metnum.lab@gmail.com`.