Aprendizaje automático

Departamento de Ingeniería en Informática
ITBA

Trabajo Práctico 3

- Para cargar datos de un archivo ascii, por ej. los datos del ej. 4.

```
load -ascii 'c:\.....\tp3\cerebros.txt'
```

- Para calcular los percentiles, z_α, $t_{n,\alpha}$

```
x = norminv(p,mu,sigma)
```
devuelve el percentil P de la distribución normal con media mu y desvío sigma (default: mu=0, sigma=1)

```
x=tinv(p,v)
```
devuelve el percentil P de la T de Student con V grados de libertad

Por ejemplo:

`x=norminv(0.95)` => $z_{0.05}$

`x=tinv(0.95,6)` => $t_{6,0.05}$

- Test t

```
[h,p,ci,stats] = ttest(x,...)
```
Realiza el test-*t* para la hipótesis nula de que los datos en el vector `x` provienen de una muestra $\mathcal{N}(0,\sigma^2)$ con σ desconocido.
Devuelve:

- `h=0:` indica que la hipótesis nula ($\mu = 0$) no puede ser rechazada a nivel α (default $\alpha = 0.05$)
  `h=1:` indica que la hipótesis nula puede ser rechazada a nivel α
- `p`: valor *p*
- `ci`: intervalo de confianza de nivel 100*(1 – α)% (default $\alpha = 0.05$)
- `tstat`: valor del estadístico del test
- `df`: grados de libertad
- `sd`: desvío estandard muestral

```
h = ttest(x,y)
```
Realiza el test-*t* para la hipótesis nula de que los datos de la diferencia entre `x-y` provienen de una muestra $\mathcal{N}(0,\sigma^2)$ con σ desconocido (test para muestras apareadas)

```
h = ttest(...,alpha,tail)
```

Realiza el test-*t* para la hipótesis alternativa especificada en `tail`, para la cual hay 3 alternativas :

- `'both'`: $\mu \neq 0$ (test a 2 colas) (default).
- `'right'`: $\mu > 0$ (test cola derecha)
- `'left'`: $\mu < 0$ (test cola izquierda)

- Para encontrar la recta de regresión

```
stats = regstats(responses,data,model,whichstats)
```

Realiza la regresión de `responses` usando los datos en la matriz `data` usando un modelo lineal multiple. Por default, agrega el término constante.
`model`: especifica cómo se crea la matriz de diseño. Algunas opciones son:

- `'linear'`: constante y términos lineales (default)
- `'purequadratic'`: constante, términos lineales y cuadráticos

Crea un output `stats` que contiene los estadísticos listados en `whichstats`. Por default devuelve todos los estadísticos. Algunas opciones son:

- `'beta'`: coeficientes de la regresión
- `'yhat'`: valores ajustados de las respuestas
- `'r'`: residuos
- `'mse'`: error cuadrático medio
- `'rsquare'`: R-cuadrado
- `'tstat'`: estadísticos t para los coeficientes

- Para ajustar polinomios:

```
p = polyfit(x,y,n)
```

Encuentra los coeficientes del polinomio p(x) de grado `n` que ajusta que mejor ajusta los datos `y` usando mínimos cuadrados.
`p` es un vector fila de largo `n+1` que contiene los coeficientes del polinomio en orden descendiente de las potencias, p(1)*x^n + p(2)*x^(n-1) +...+ p(n)*x + p(n+1).

```
y = polyval(p,x)
```

Devuelve el valor del polinomio evaluado en `x`.
`p` es un vector fila de largo `n+1` que contiene los coeficientes del polinomio en orden descendiente de las potencias, p(1)*x^n + p(2)*x^(n-1) +...+ p(n)*x + p(n+1).

- Para encontrar la matriz de correlación muestral

```
r=corrcoef(x)
```

Calcula la matriz R de los coeficientes de correlación para un array `x`, en el cual cada fila es una observación y cada columna una variable.

- Para hacer la selección de variables

```
[b,se,pval,inmodel,stats,nextstep,history]=stepwisefit(x,y,'param1',val1,
'param2',val2,...)
```

Usa regresión stepwise para encontrar el modelo lineal que ajusta la variable respuesta `y` como función de las variables predictoras representadas por las columnas de la matriz `x`.
Devuelve:

- `b:` vector de los valores estimados de los coeficientes
- `se:` vector de errores standard de `b`
- `pval:` vector de los valores *p* para testear si `b` es 0
- `inmodel:` vector lógico que indica qué predictores están en el modelo final
- `stats:` contiene varios estadísticos
- `nextstep:` recomienda cuál sería el próximo paso: qué predictor entraría o saldría del modelo
- `history:` contiene información acerca de los pasos

Algunos de los parámetros que se pueden dar como input son:

- `'inmodel':` un vector lógico o una lista de los números de columnas que indican qué predictores se incluyen en el modelo inicial (default: ninguno)
- `'penter'`: Max valor *p* para agregar un predictor al modelo (default 0.05)
- `'premove'` Min valor *p* para remover un predictor del modelo (default 0.10)

- Componentes principales

```
[coeff, score, latent] = princomp(x)
```

Realiza el análisis de componentes principales para la matriz de datos `x` de n x p, en el cual cada fila es una observación y cada columna una variable.

Devuelve:

- `coeff:` una matriz de p x p, donde cada columna contiene los coeficientes de cada una de las componentes principales. Las columnas están en orden decreciente de acuerdo a la varianza de la componente.
- `score:` la representación de `x` en el espacio de las componentes principales (autovectores). Las filas corresponden a las observaciones y las columnas a las componentes
- `latent:` los autovalores de `x`.

```
cumsum(latent)./sum(latent)
```

Calcula el porcentaje de varianza explicada por cada componente.

```
biplot(coefs,'scores', scores, 'varlabels',varlabs)
```

Realiza un plot simultáneo de `coefs` y de `scores`. Cada observación (fila de `scores`) está representado por un punto en el biplot.
Los `scores` corresponden a los autovectores obtenidos con las componentes principales.