VARIOUS: `install.packages("pkgName")` , `library("dplyr")` , `? funcName` , `rm(list = ls())` , `setwd("wDir")` , `NA` , `NULL` , `rep(NA, 1000)` , `sample(myvector, 100,replace=TRUE)` , `x[c(-2, -10)]` , `x[!is.na(x)]` , `vect = c(foo = 11, bar = 2, norf = NA)` , `names(vect)` , `length(my_vector)` , `dim(my_vector) <- c(4, 5)` , `as.vector(m)` , `m = matrix(data = c(1:20), nrow = 5, ncol = 4, byrow = FALSE, dimnames = list(list("a","b","c'` , `cbind(patients, my_matrix)` , `contains(match, ignore.case = TRUE)`

DF: `df <- data.frame(patients, m)` , `df <- read.csv("myfile.csv")` , `colnames(my_data) <- cnames` , `dim(df)` , `nrow(df)` , `ncol(df)` , `df["a",]` , `df[,c("A","C")]` or `df$A` , `str(df)`

PLOT/VISUAL:

`plot(x=df$col1,y=df$col2,xlab="X", ylab="Y", col=2, pch=1, xlim = c(20, 40))` , `boxplot(formula, dfsource,...)` , `hist(vector)` , `View(df)`

```
ggplot(df) +
  geom_histogram(data=df, aes(col1, ..density..), fill="white", color="darkred", binwidth=0.5) +
  geom_density(kernel="gaussian", aes(col1), bw=0.5) +
  stat_ecdf(data=df, aes(col1), color="darkblue")
```

DATAPROC: `my_tbldf <- tibble::as_tibble(mydf)` , select colmumns: `select(my_tbldf, col1, col2, ...)` , select rows: `filter(my_tbldf, col1=="something", col2<="somethingElse", grepl("matchToFind",col3))` , `arrange(my_tbldf,sortCol1, sortCol2, desc(sortCol3))` , `mutate(my_tbldf, newCol = [some logic based on other columns' values], newCol2 = [here we can` , `mydf$varC[(!is.na(mydf$varB)) & (mydf$varA == mydf$varB)] <- 1` , `mydf$varC <- ifelse(mydf$varA == mydf$varC, 1, 0)` , `summarize(my_tbldf, c1_avg = mean(col1), c1_sum=sum(col1), c1_nDist=n_distinct(col1), rec_coun` , `my_tbldf %>% group_by(catVar) %>% summarize(count = n(),unique = n_distinct(ip_id),...)` , wide to long: `gather(my_tbldf, nameToGiveToNewKeyCol, nameToGiveToNewValueCol, namesOldColumnsFromWhichToTak` , `separate(my_tbldf, colToSeparate, namesToGiveToSeparatedCols, separator)` , long to wide: `spread(my_tbldf, colWhoseUniqueValuesWillBecomeCols, colStoringTheValue )` , `unique(my_tbldf)` , vstack: `bind_rows(my_tbldf1,my_tbldf2,...)` , hstack: `bind_cols(my_tbldf1,my_tbldf2,...)`

REGR: `myFit <- lm(dep_var ~ a_var + other_var, data=subset(mydf,...))` , `summary(myFit)` , `coefficients(myFit)` , `confint(myFit, level=0.9)` , `residuals(myFit)` , `SRR = anova(myTest)$"Sum Sq"` , `linearHyphotesis(model)` , `rdd.{DCdensity.RDestimate}` package: *Regression Discontinuity Designs*

PHYTON

NP: `argmax` , `arange(from,to,by)` , `zeros`

DISTRIBUTIONS (Python: `from scipy import stats` , R: `library(extraDistr)`):

DISCRETE DISTRIBUTIONS (Python - `stats.[distributionName]` , R) : Discrete uniform `randint(lRange,uRange)` `dunif(lRange,uRange)` , Bernoulli `bernoulli(p)` `bern(p)` , Binomial `binom(n,p)` `binom(n,p)` , Categorical Not Av. `cat(ps)` , Multinomial `multinomial(n, ps)` `mnom(n,ps)` , Geometric `geom(p)` `geom(p)` , Hypergeometric `hypergeom(nS+nF,nS,nTrials)` `hyper(nS, nF, nTrias)` ,Mv hypergeometric `multivariate_hypergeom(initNByCat,nTrials)` `mvhyper(initialNByCat,nTrials)` , Poisson `poisson(rate)` `pois(rate)` , Negative Binomial `nbinom(nSucc,p)` `nbinom(nSucc,p)`

CONTINUOUS DISTRIBUTIONS (Python - `stats.[distributionName]` , R) : Uniform `uniform(lRange,uRange)` `unif(lRange,uRange)` , Exponential `expon(rate)` `exp(rate)` , Laplace `laplace(loc,scale)` `laplace(loc,scale)` , Normal `norm(μ,math.sqrt(σsq))` `norm(μ,sqrt(σsq))` , Erlang `erlang(n,rate)` Use gamma, Cauchy `cauchy(μ, σ)` `cauchy(μ,σ)` , Chisq `chi2(df)` `chisq(df)` , T Dist `t(df)` `t(df)` , F Dist `f(df1, df2)` `f(df1,df2)` , Beta Dist `beta(shapeα,shapeβ)` `beta(shapeα,shapeβ)` , Gamma Dist `gamma(shapeα,1/rateβ)` `gamma(shapeα,1/rateβ)`

- **Sample**: `d.rvs()` , `r[distributionName](1,distributionParameters)` , e.g. `runif(1,10,20)`
- **Quantiles** $(F^{-1}(y))$ with $y = CDF(x)$: `d.ppf(y)` , `q[distributionName](y, distributionParameters)` , e.g. `qunif(0.2,10,20)`
- **PDF/PMF**: `d.pmf(x)` for discrete r.v. and `d.pdf(x)` for continuous ones, `d[distributionName](x, distributionParameters)` , e.g. `dunif(15,10,20)`
- **CDF**: `d.cdf(x)` , `p[distributionName](x, distributionParameters)` , e.g. `punif(15,10,20)`

Kernel estimation of a PMF from data: $\hat{f}_b(x) = \frac{1}{nb}\sum_{n=1}^{1} K(\frac{x-x_i}{b})$
Stoc dominance of X over Y: $P(X \geq t) \geq P(Y \geq t)\ \forall t$
Randomised Control Trials (RCT): Unit/Action{Treatment,Control}/Outcome Completely randomised /Stratified / Pairwise randomised / Clustered randomised / Pairwise randomised
SUTVA (Stable Unit Treatment Value Assumption): The potential outcomes for any unit do not vary with the treatments assigned to other units; and, for each unit, there are no different forms or versions of each treatment unit leading to different outcomes
$E[Y_i^{obs}|W_i = 1] - E[Y_i^{obs}|W_i = 0] = E[Y_i(1)|W_i = 1] - E[Y_i(0)|W_i = 0] = E[Y_i(1)|W_i = 1] - E[Y_i(0)|W_i = 1] + E[Y_i(0)|W_i = 1] - E[Y_i(0)|W_i = 0]$ where the first diff is the **treatment effect** of the units actually treated (what we want) and the second diff is the selection bias, the diff in **potential outcome** of not being treated between the units effectivelly treated and those not actually treated. $E[Y_i(0)|W_i = 0]$ is the expected value related to the potential outcome of not being treated for a unit that has been effectivelly not being treated (observed)
$CI = [\bar{X}_n - \Phi^{-1}(1-\frac{\alpha}{2})\frac{\sigma}{\sqrt{n}}; \bar{X}_n + \Phi^{-1}(1-\frac{\alpha}{2})\frac{\sigma}{\sqrt{n}}]$
Sharp null in RCT: treatement no effect ∀ units, not just on average
Fisher exact test: (1) for each possible permutation of units treated compute the "would be observed" treatment effect under the null $Y(0)$ = $Y(1)$ (2) compute the p-value as the share of permutation whose abs treatment effect > observed one
R²: $R^2 = 1 - \frac{SSR}{SST} = 1 - \frac{SSR}{SSR+SSM} = 1 - \frac{\sum_i (Y_i - \hat{Y}_i)^2}{\sum_i (Y_i - \bar{Y}_n)^2} = 1 - \frac{\sum_i (Y_i - \hat{Y}_i)^2}{\sum_i (Y_i - \hat{Y}_i)^2 + \sum_i (\hat{Y}_i - \bar{Y}_n)^2}$

F-test of overall significance: $T = \frac{n-d}{d-1}\frac{R^2}{1-R^2} \sim F(d-1, n-d)$ (reject H_0: all beta except intercept = 0?)

Multiple hyphotesis test: $\|T_n^{(d)} = \frac{u_d^T\hat{\beta}-u_d^T\beta_0}{\sqrt{\hat{\sigma}^2 u_d^T(X^TX)^{-1}u_d}}\|^2 > q_{\chi^2_{N-D}(\alpha)}$ or $H_O : R^T\beta = c$ (with R a D-dims col vector and c a scalar): $T = \frac{R^T\hat{\beta}-c}{\sqrt{\hat{\sigma}^2 R^T(X^TX)^{-1}R}} > q_{TDIST_{N-D}(\alpha)}$

Restricted model: H_0: restrictions are real. (1) estimate the unrestricet model; (2) impose restriction and estimate the restricted mode; (3) compute the test statistic $T = \frac{\frac{SSR_R - SSR_U}{r}}{\frac{SSR_U}{n-(d+1)}}$ with r the number of restricitons; (4) Reject H_0 if $T > FDIS^{-1}_{r,n-(d+1)}(1-\alpha)$

Diff in diff: $Y = \alpha + \beta D\{\text{0: pre treatement; 1:post}\} + \gamma M\{\text{O: control group; 1:treated group}\} + \sigma M * D + \epsilon$ Interpretation: α: non affected group average before treatment, β: non affected group post vs pre treatment, γ: pre treatment initial diff between treated and untreated (control) group, σ diff between pre/post treatment diff in the treated group vs control one

Interaction terms: (a) vector of all possible combination treatment/control dummies to test treatment effect on each group (b) model with dummy for control group (αD) + combinations of dummies with continuous vars on treatment intensity (βX): α returns the shift effect of the dummy, β return the change in slope effect of dummy on the X.

Fixed effect regression: dummies for systematic difference that doesn't vary over time between control/treated group, one 0/1 dummy set for categorical variable of interest, plus interaction term between control/treated group and treatment intensity

Log models: log(X) → log(Y): β is the elasticity; X → log(Y): relative variation in Y on a unit variation in X

Box cox transformation: $Y = \frac{1}{X\beta}$ → estimate as $\frac{1}{Y} = X\beta$

Discrete choice model: $P = \frac{e^{X\beta}}{1-e^{X\beta}}$ → estimate as $Y = \ln(\frac{P}{1-P}) = X\beta$

plane: From normal and point to algebric coordinated: $(\vec{v}, x_p) \rightarrow (\theta, \theta_0) : \theta = \vec{v}, \theta_0 = -\vec{v} \cdot x_p$

Perceptron

- if $y^{(i)}(\theta \cdot x^{(i)} + \theta_0) \leq 0$ then
  - update $\theta = \theta + y^{(i)}x^{(i)}$
  - update $\theta_0 = \theta_0 + y^{(i)}$

Margin boundaries x : $\frac{\theta}{||\theta||} \cdot x + \frac{\theta_0}{||\theta||} = \frac{k}{||\theta||}$

```
huberLoss(x) = x >= 1 ? 0 : 1 - x
```

Support Vector Machine (SVM): $min_{\theta,\theta_0} J(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^n \text{HubLoss}_h(y^{(i)} * (\theta \cdot x^{(i)} + \theta_0)) + \frac{\lambda}{2} \mid\mid \theta \mid\mid^2$

SGD update rule of SVM models: $\theta = \theta - \eta_t * \nabla J_i(\theta) \Rightarrow \theta = \theta - \frac{1}{1+t} *$

$\left(\begin{cases} 0 & \text{when loss=0} \\ -y^i\mathbf{x}^{(i)} & \text{when loss >0} \end{cases} + \lambda\theta\right)$

```
linear_model.SGDClassifier(loss='hinge', penalty='l2', alpha=alpha[i])
ms.cross_val_score(model, X, y, cv=5)
```

Ridge regression: $J_{n,\lambda} = R_n(\theta) + \frac{\lambda}{2}||\theta||^2 = \frac{1}{n}\sum_{t=1}^n \frac{(y^{(t)} - \theta \cdot \mathbf{x}^{(t)})^2}{2} + \frac{\lambda}{2}||\theta||^2$

SGD update rule for ridge regression: $\theta = \theta - \eta * \nabla_\theta = (1 - \eta\lambda)\theta + \eta * (y^{(t)} - \theta \cdot \mathbf{x}^{(t)}) * \mathbf{x}^{(t)})$

Kernel function: $k(x, x'; \phi) \in \mathbb{R}^+ = \phi(x) \cdot \phi(x')$, e.g. $K(x, x') = e^{-\frac{1}{2}||x-x'||^2}$

Kernel perceptron:

- α = 0 # initialisation of the vector
- if y^i $\sum_j[\alpha^j y^j k(x^j,x^i)] \leqq 0$ # checking if prediction is right
  - α^i += 1 # update α^i if mistake

predictions: $\hat{y}^{(i)} = (\sum_{j=1}^n \alpha^{(j)}y^{(j)}k(x^{(j)}, x^{(i)}) > 0) * 2 - 1$

Collaborative filtering: $X = UT^T$, $J(\mathbf{u}, \mathbf{v}; Y, \lambda) = \frac{\sum_{a,i \in D}(Y_{a,i} - u_a * v_i)^2}{2} + \frac{\lambda}{2}\sum_a^n u_a^2 + \frac{\lambda}{2}\sum_i^m v_i^2$

$\text{Softmax}(\mathbf{Z} \in \mathbb{R}^K) \in \mathbb{R}^{+K} = \frac{1}{\sum_{j=1}^k e^{Z_j}} * e^{\mathbf{Z}}$

SGD update rule for NN: $w_{i,j}^l \leftarrow w_{i,j}^l - \eta * \frac{\partial \mathcal{L}(f(X;W),Y)}{\partial w_{i,j}}$

RNN: $g_t = \text{sigmoid}(W^{g,s}s_{t-1} + W^{g,x}x_t) = \frac{1}{1+e^{-(W^{g,s}s_{t-1}+W^{g,x}x_t)}}$ $s_t = (1 - g_t) \odot s_{t-1} + g_t \odot \tanh(W^{s,s}s_{t-1} + W^{s,x}x_t)$

K-means: (1) assign points to the closest representative $cost(z^{(1)}, ..., z^{(j)}, ..., z^{(Z)}) = \sum_{i=1}^n min_{j=1,...,K}||x^{(i)} - z^{(j)}||^2$; (2) find the best representative given its group: cost(C_1,..,C_j,...,C_K) = min_{z{(1)},...,z{(j)},...,z{(Z)}}\sum_{j=1}K \sum_{i \in C_J} ||x^{(i)} - z{(j)}||2$

GMM/EM algorithm

- likelihood: $L(S_n) = \prod_{i=1}^n \sum_{j=1}^K p_j N(x_i; \mu_j, \sigma_j^2)$
- E-step: $p(J = j|X = x_i) = \frac{p_j * N(x_i;\mu_j,\sigma_j^2)}{\sum_{j=1}^K p_j * N(x_i;\mu_j,\sigma_j^2)}$
- M-step: FOC in terms of two parameters μ_j, σ_j^2 to miinimise the likelihood:
  - $\hat{n}_j$, the number of points of each cluster: $\hat{n}_j = sum_{i=1}^n p(j|i)$ (this is actually a weighted sum of the points, where the weights are the relative probability of the points to belong to cluster j)
  - $\hat{p}_j = \frac{\hat{n}_j}{n}$
  - $\hat{\mu}^{(j)} = \frac{1}{\hat{n}_j}\sum_{i=1}^n p(j|i) * x^{(i)}$
  - $\hat{\sigma}_j^2 = \frac{1}{\hat{n}_j d}\sum_{i=1}^n p(j|i) * ||x^{(t)} - \mu^{(j)}||^2$

Bellman value function:

- $V^*(s) = \max_a Q^*(s, a) = Q^*(s, a = \pi^*(s))$ where π^* is the optimal policy.

- $V^*(s) = \max_a \sum_{s'} T(s,a,s')(R(s,a,s') + \gamma V^*(s'))$

Q value interation

- $V^*(s_k) = \max_a \sum_{s\prime} T(s,a,s')(R(s,a,s') + \gamma V^*(s'_{k-1}))$
- $\pi^*(s) = argmax_a Q^*(s,a) = argmax_a \sum_{s'} T(s,a,s')(R(s,a,s') + \gamma V^*(s'))$

Estimated state/reward matrices:

- $\hat{T}(s,a,s') = \frac{\text{count}(s,a,s')}{\sum_{s''} \text{count}(s,a,s'')}$
- $\hat{R}(s,a,s') = \frac{\sum_{t=1}^{\text{count}(s,a,s')} R_t(s,a,s')}{\text{count}(s,a,s')}$ (the average of the rewards we collect)

Q value by sampling:

- $Q^*(s,a) = \sum_{s'} T(s,a,s') \left(R(s,a,s') + \gamma \max_{a'} Q^*(s',a') \right)$
- $Q^*(s,a) = \frac{1}{k} \sum_{i=1}^{k} Q_i^*(s,a) = \frac{1}{k} \sum_{i=1} \left(R(s,a,s'_i) + \gamma \max_{a'} Q^*(s'_i,a') \right)$

The Q-value for (s,a) at sample $i+1$ would then be equal to

$$Q_{i+1}(s,a) = (1-\alpha)Q_i(s,a) + \alpha * sample_i(s,a)$$
$$Q_{i+1}(s,a) = Q_i(s,a) - \alpha(Q_i(s,a) - sample_i(s,a))$$
$$Q_{i+1}(s,a) = Q_i(s,a) - \alpha(Q_i(s,a) - (R(s,a,s'_i) + \gamma \max_{a'} Q^*(s'_i,a')))$$