Comparision of various ML algorithms

Algorith Name	Library	Function	Calculation Hint	Evaluation Measures
		Classificati	on	
KNN	from sklearn.neighb ors import KNeighborsClas sifier	KNeighborsClass ifier(n_neighbor s = 2)		 Accuracy Kappa Value Precision Recall Sensitivity Specificity F-1 Score Error rate
Decision Tree	from sklearn.tree import DecisionTreeCl assifier	DecisionTreeCla ssifier()	EntropyInformationGain	 Accuracy Kappa Value Precision Recall Sensitivity Specificity F-1 Score Error rate
Random Forest		RandomForestCl assifier(n_estim ators=100)	-	 Accuracy Kappa Value Precision Recall Sensitivity Specificity F-1 Score Error rate
Support Vector Machine	from sklearn import svm	svm.SVC(kernel ="linear")	-	 Accuracy Kappa Value Precision Recall Sensitivity Specificity F-1 Score Error rate
Naïve Bayes	from sklearn.naive_b ayes import GaussianNB	GaussianNB()	frequency table, cumulative probability, probability normalization	 Accuracy Kappa Value Precision Recall Sensitivity Specificity F-1 Score

		• •		• Error rate
		rvised Learning	: Regression	
Linear	from sklearn.linear_ model import LinearRegressio n	LinearRegressio n()	-	R SquaredMean Absolute ErrorMean Squared
Logistic	from sklearn.linear_ model import LogisticRegressi on	LogisticRegressi on()	-	 R Squared Mean Absolute Error Mean Squared Accuracy Kappa Value Precision Recall Sensitivity Specificity F-1 Score
	Unsup	ervised Learnin	g: Clustering	
K – means	from sklearn.cluster import KMeans	KMeans(n_clust ers=n_clusters, init=init, n_init=n_init)	 Euclide an distanc e betwee n data points and Centroi ds Assign clusters based on minimu m distanc e Re-comput e new clusters by taking the 	

K - Medoids from sklearn_extra.c luster import KMedoids Hierarchical • from sklearn.c luster import Agglome rativeClu stering • from scipy.clu ster import hierarch y Apriori from from Agriori • Identify • Identify • Manhat tan distanc • Compar e new and previou s cost • If new cost is lesser, swap otherwi se undo swap • Euclide an distanc e • Data scaling distanc e • Derieve the distanc e matrix • Group the items based on smallest distanc e • Apriori from • Identify • Calcula				mean	
K - Medoids from sklearn_extra.c luster import KMedoids KMedoids KMedoids KMedoids KMedoids KMedoids KMedoids Compar e new and previou s cost is lesser, swap otherwi se undo swap Hierarchical From sklearn.c luster import Agglome rativeClu stering From scipy.clu stering From sklearn.c luster From sklearn.c From sklearn.c					
K – Medoids from sklearn_extra.c luster import KMedoids Hierarchical **From sklearn.c luster import Agglome rativeClustering of from scipy.clustering of from scipy.clus					
sklearn_extra.c luster import KMedoids Hierarchical * from sklearn.c luster import Agglome rativeClu stering * from scipy.clu ster import hierarch y * Mariori * from sklearn.c luster import Agglome rativeClu stering * from scipy.clu ster import hierarch y * Identify * Identify * Identify * Compar e new and previou s cost is lesser, swap otherwi se undo swap * Euclide an distanc e * Data scaling distanc e * Derieve the distanc e * Derieve the distanc e * Group the items based on smallest distanc e * Apriori * Identify * Calcula * Compar e new and previou s cost is lesser, swap otherwi se undo swap * Euclide an distanc e * Derieve the distanc e * Orompar e new and previou s cost is lesser, swap otherwi se undo swap * Normaliz e data * Derieve the distanc e * Orompar e new and previou s cost is lesser, swap otherwi se undo swap * Normaliz e data * Derieve the distanc e * Orompar e new and previou s cost is lesser, swap otherwi se undo swap * Normaliz e data * Derieve the distanc e * Orompar e new and previou s cost is lesser, swap otherwi se undo swap * Normaliz e data * Derieve the distanc e * Orompar e new and previou s cost is lesser, swap otherwi se undo swap * Normaliz e data * Derieve the distanc e * Orompar e new and previou s cost is lesser, swap otherwi se undo swap * Normaliz e data * Derieve the distanc e * Orompar e new and previou s cost is lesser, swap otherwi se undo swap * Derieve the distanc e * Orompar e new and previou s cost is lesser, swap otherwi se undo swap * Normaliz e data * Derieve the distanc e * Orompar e new and previou solution is not previou so	K – Medoids	from	KMedoids(n clu	_	•
KMedoids Particial Partic			_		
Hierarchical • from sklearn.c luster import Agglome rativeClu stering • from scipy.clu ster import hierarch y • from ster import hierarch y • from scipy.clu ster import hierarch y • from scipy.clu ster import hierarch y • from scipy.clu ster import hierarch y • Identify • Compar e new and previou s • If new cost is lesser, swap otherwi se undo swap • Euclide an distanc e • Derieve the distanc e • Derieve the distanc e • Group the items based on smallest distanc e • Apriori from • Identify • Calcula •				distanc	
Hierarchical		KMedoids			
Hierarchical • from sklearn.c luster import Agglome rativeClu stering • from scipy.clu ster import hierarch y Apriori from • Identify • Identify • Identify • Calcula • If new cost is lesser, swap otherwi se undo swap • Euclide an distanc e • Derieve the distanc e • Derieve the distanc e • Group the items based on smallest distanc e • Apriori from • Identify • Calcula				_	
Hierarchical • from sklearn.c luster import Agglome rativeClu stering • from scipy.clu ster import hierarch y • Identify • Calcula •					
Hierarchical • from sklearn.c luster import Agglome rativeClu stering • from scipy.clu ster import hierarch Apriori • If new cost is lesser, swap otherwi se undo swap • Data scaling • Euclide an distanc e data • hierarchy linkage(sprincipal of the items based on smallest distanc e • Group • Group • Group • Group • Group • He items • based • On smallest distanc • On smallest distanc • Calcula					
Hierarchical • from sklearn.c luster import Agglome rativeClu stering • from scipy.clu ster import hierarch y • from scipy.clu ster import hierarch y Apriori from • Identify • Laculide scaling an distanc e • Buclide an distanc e • Derieve the distanc e • matrix • Group the items based on smallest distanc e Apriori				s cost	
Hierarchical • from sklearn.c luster import Agglome rativeClu stering • from scipy.clu ster import hierarch y • Apriori from • Identify • Calcula • Identify • Data scaling an distanc e • Derieve the distanc e • Derieve the items based on smallest distanc e					
Hierarchical • from sklearn.c luster import Agglome rativeClu stering • from scipy.clu ster import hierarch y Apriori • from shlearn.c luster scaling e data scaling distanc e e data e hierarchy principal distanc e matrix • hierarchy ser import hierarch y • Identify • Calcula • Euclide an distanc e e Derieve the distanc e e matrix • Group the items based on smallest distanc e					
Hierarchical • from sklearn.c luster import Agglome rativeClu stering of from scipy.clu ster import hierarch y • Mormaliz distanc e data e hierarchy linkage(stering of the scipy.clu ster import hierarch y • Mormaliz distanc e data e e hierarchy the distanc e matrix • Group the items based on smallest distanc e • Apriori from • Identify • Calcula •				,	
Hierarchical • from sklearn.c luster import Agglome rativeClu stering • from scipy.clu ster import hierarch y • Apriori from • Identify • Data scaling an distanc e • Derieve the distanc e • Derieve the distanc • Group the items based oon smallest distanc e • Apriori from • Identify • Calcula				_	
Hierarchical • from sklearn.c luster import Agglome rativeClu stering • from scipy.clu ster import hierarch y • Apriori from • Data scaling an distanc e • Normaliz e data e • hierarchy linkage(sthe the distanc e • hierarchy cluster the distanc e • Group the items based on smallest distanc e • Apriori from • Identify • Calcula				se undo	
sklearn.c luster import e data e Agglome rativeClu stering of from scipy.clu ster import hierarch y Apriori from luster import luster import from sklearn.c luster import hierarch g luster import luster import hierarch g luster import luster items luste				swap	
luster import e data e data e Derieve rativeClu stering or from scipy.clu ster import hierarch y	Hierarchical				•
import Agglome rativeClu stering • from scipy.clu ster import hierarch y Apriori e data • hierarchy .linkage(principal Data, method= "ward") • Group the items based on smallest distanc e Apriori • Identify • Calcula • Derieve the distanc e • Derieve the distanc e • Data, matrix • Group the items based on smallest distanc e			_		
Agglome rativeClu stering • hierarchy .linkage(the distanc e matrix ster import hierarch y					
rativeClu stering principal distanc • from Data, e matrix ster "ward") import hierarch y Apriori from • Identify • Calcula					
• from scipy.clu method= matrix ster "ward") the items based on smallest distanc e Apriori from • Identify • Calcula •				the	
scipy.clu ster "ward")		_			
ster "ward") • Group the items based on smallest distanc e Apriori from • Identify • Calcula •			· ·		
import the items based on smallest distanc e Apriori from • Identify • Calcula •					
hierarch y based on smallest distanc e Apriori from • Identify • Calcula •			waru)	_	
on smallest distanc e Apriori from • Identify • Calcula •					
smallest distanc e Apriori from • Identify • Calcula •		у		based	
Apriori from • Identify • Calcula •					
Apriori from • Identify • Calcula •					
Apriori from • Identify • Calcula •					
	Apriori	from	Identify		•
(Association mlxtend.freque item set te the		mlxtend.freque			
	rule)	-	_		
import apriori, values) count					
association_rul • Identify for each		_	· ·		
on item		Co			
• apriori(o set					

he_df,	• Combin
min_sup	e the
port=0.2,	item
use_coln	sets as
ames=Tru	per
е,	provide
verbose=	d
1)	thresh
 associatio 	hold
n_rules(fr	value
eq_items,	• Calcula
metric="c	te the
onfidence	confide
",	nce for
min_thre	all
shold=0.6	possible
)	combin
	ation