### Topics

+ review song similarity; print result in HTML

+ review movie similarity

+ scalable/indexed search engine

+ kNN classification

+ kNN clustering

# Reviewing similarity functions

# Scalable search engine

# kNN classification

|  |
| --- |
| **public** **static** **void** main(String[] args)  {  Classifier<**double**[], String> c = **new** MyClassifier();  c.train(**new** **double**[] { 100, 0, 0 }, "red");  c.train(**new** **double**[] { 101, 0, 0 }, "red");  c.train(**new** **double**[] { 102, 0, 0 }, "red");  c.train(**new** **double**[] { 105, 0, 0 }, "red");    c.train(**new** **double**[] { 0, 301, 0 }, "green");  c.train(**new** **double**[] { 0, 309, 0 }, "green");  c.train(**new** **double**[] { 0, 304, 0 }, "green");  c.train(**new** **double**[] { 0, 305, 0 }, "green");  c.train(**new** **double**[] { 0, 307, 0 }, "green");    c.train(**new** **double**[] { 0, 0, 290 }, "blue");  c.train(**new** **double**[] { 0, 0, 291 }, "blue");  c.train(**new** **double**[] { 0, 0, 293 }, "blue");  c.train(**new** **double**[] { 0, 0, 298 }, "blue");    String res = c.predict(**new** **double**[] { 105, 0, 0 });  System.***out***.println(res);    res = c.predict(**new** **double**[] { 120, 0, 0 });  System.***out***.println(res);    String res1 = c.predict(**new** **double**[] { 100, 301, 0 });  System.***out***.println(res1);  } |
| **public** **abstract** **class** Classifier<S, T>  {  **private** Map<S, T> examples = **new** LinkedHashMap<S, T>();  **public** **void** train(S sk, T tk)  {  examples.put(sk, tk);  }  **public** T predict(S q)  {  **double** dres = -Double.***MAX\_VALUE***;  T tres = **null**;    **for**(S nk: examples.keySet())  {  **double** dk = jaccardSimilarScore(nk, q);  **if**(dk > dres) { dres = dk; tres = examples.get(nk); }    System.***out***.println(dk + ":" + nk);  }    **return** tres;  }  **public** **abstract** **double** jaccardSimilarScore(S nk, S q);  } |
| **public** **class** ClaireSearchEngineSong **extends** ClaireSearchEngine<ClaireSong> {  **private** ClaireSongMatcher matcher = **new** ClaireSongMatcher();    @Override  **public** **double** jaccardIndex(ClaireSong sk, ClaireSong q) **throws** Exception  {  **return** matcher.jaccardSimilarScore(sk, q);  }  } |