DBMS\_PREDICTIVE\_ANALYTICS.EXPLAIN (

data\_table\_name IN VARCHAR2,

explain\_column\_name IN VARCHAR2,

result\_table\_name IN VARCHAR2,

data\_schema\_name IN VARCHAR2 DEFAULT NULL);

BEGIN

DBMS\_PREDICTIVE\_ANALYTICS.EXPLAIN(

data\_table\_name => 'Beer\_data',

explain\_column\_name => 'overall',

result\_table\_name => 'explain\_overall');

END;

/

select attribute\_name, explanatory\_value from explain\_overall order by rank;

CREATE VIEW beer\_view AS

SELECT IDINDEX, STYLE, BREWERID, ABV, OVERALL

FROM Beer\_data;

DECLARE

p\_accuracy NUMBER(10,9);

BEGIN

DBMS\_PREDICTIVE\_ANALYTICS.PREDICT(

accuracy => p\_accuracy,

data\_table\_name =>'Beer\_view',

case\_id\_column\_name =>'idindex',

target\_column\_name =>'overall',

result\_table\_name => 'Predict\_overall');

DBMS\_OUTPUT.PUT\_LINE('Accuracy: ' || p\_accuracy);

END;

/

BEGIN

DBMS\_PREDICTIVE\_ANALYTICS.PROFILE(

DATA\_TABLE\_NAME => 'Beer\_view',

TARGET\_COLUMN\_NAME => 'overall',

RESULT\_TABLE\_NAME => 'Overall\_profile');

END;

/

DECLARE

summary\_values DBMS\_STAT\_FUNCS.SummaryType;

significance number;

BEGIN

DBMS\_STAT\_FUNCS.SUMMARY(

p\_ownername => 'HELIML',

p\_tablename => 'STUDENT\_ENROLLMENT',

p\_columnname => 'AGE',

p\_sigma\_value => 3,

s => summary\_values);

dbms\_output.put\_line('Summary statistics: ');

dbms\_output.put\_line('Number of records: '

||summary\_values.count);

dbms\_output.put\_line('Min value: '||summary\_values.min);

dbms\_output.put\_line('Max value: '||summary\_values.max);

dbms\_output.put\_line('Variance: '

||round(summary\_values.variance));

dbms\_output.put\_line('Stddev: '||round(summary\_values.stddev));

dbms\_output.put\_line('Mean: '||summary\_values.mean);

dbms\_output.put\_line('Mode: '||summary\_values.cmode(1));

dbms\_output.put\_line('Median: '||summary\_values.median);

dbms\_output.put\_line('Quantiles');

dbms\_output.put\_line('1st Quantile: '

||summary\_values.quantile\_5);

dbms\_output.put\_line('2nd Quantile: '

||summary\_values.quantile\_25);

dbms\_output.put\_line('3rd Quantile: '

||summary\_values.quantile\_75);

dbms\_output.put\_line('4th Quantile: '

||summary\_values.quantile\_95);

dbms\_output.put\_line('Extreme count: '

||summary\_values.extreme\_values.count);

dbms\_output.put\_line('Top Five Values: '

||summary\_values.top\_5\_values(1)||','

||summary\_values.top\_5\_values(2)||','

||summary\_values.top\_5\_values(3)||','

||summary\_values.top\_5\_values(4)||','

||summary\_values.top\_5\_values(5));

dbms\_output.put\_line('Bottom Five Values: '

||summary\_values.bottom\_5\_values(1)||','

||summary\_values.bottom\_5\_values(2)||','

||summary\_values.bottom\_5\_values(3)||','

||summary\_values.bottom\_5\_values(4)||','

||summary\_values.bottom\_5\_values(5));

dbms\_output.put\_line('Normality test');

DBMS\_STAT\_FUNCS.normal\_dist\_fit(

ownername => 'HELIML',

tablename => 'STUDENT\_ENROLLMENT',

columnname => 'AGE',

test\_type => 'SHAPIRO\_WILKS',

mean => summary\_values.mean,

stdev => summary\_values.stddev,

sig => significance);

END;

/

select s.studentid, c.coursename, c.credits

from students s, courses c, coursespassed cp

where s.studentid=cp.studentid and

c.courseid=cp.courseid;

create view student\_credits\_view as

(select s.studentid, c.coursename, c.credits

from students s, courses c, coursespassed cp

where s.studentid=cp.studentid and

c.courseid=cp.courseid);

CREATE view students\_credit\_trans\_nested AS

SELECT studentid,

CAST(COLLECT(DM\_NESTED\_NUMERICAL

(coursename, credits)) AS DM\_NESTED\_NUMERICALS) studentcredits

FROM STUDENT\_CREDITS\_VIEW

GROUP BY studentid;

select \* from students\_credit\_trans\_nested;

CREATE TABLE Beer\_settings\_DT (

setting\_name VARCHAR2(30),

setting\_value VARCHAR2(4000));

CREATE TABLE Beer\_settings\_NB (

setting\_name VARCHAR2(30),

setting\_value VARCHAR2(4000));

CREATE TABLE Beer\_settings\_SVM (

setting\_name VARCHAR2(30),

setting\_value VARCHAR2(4000));

INSERT INTO Beer\_settings\_DT VALUES

(dbms\_data\_mining.algo\_name, dbms\_data\_mining.algo\_decision\_tree);

BEGIN

DBMS\_DATA\_MINING.CREATE\_MODEL(

model\_name => 'Beer\_DT',

mining\_function => dbms\_data\_mining.classification,

data\_table\_name => 'Beer\_training\_data',

case\_id\_column\_name => 'IDIndex',

target\_column\_name => 'Overall',

settings\_table\_name => 'Beer\_settings\_DT');

END;

/

BEGIN

DBMS\_DATA\_MINING.CREATE\_MODEL(

model\_name => 'Beer\_NB',

mining\_function => dbms\_data\_mining.classification,

data\_table\_name => 'Beer\_training\_data',

case\_id\_column\_name => 'IDIndex',

target\_column\_name => 'Overall',

settings\_table\_name => 'Beer\_settings\_NB');

END;

/

BEGIN

DBMS\_DATA\_MINING.CREATE\_MODEL(

model\_name => 'Beer\_SVM',

mining\_function => dbms\_data\_mining.classification,

data\_table\_name => 'Beer\_training\_data',

case\_id\_column\_name => 'IDIndex',

target\_column\_name => 'Overall',

settings\_table\_name => 'Beer\_settings\_SVM');

END;

/

EXEC dbms\_data\_mining.rename\_model('Beer\_DT\_current', 'Beer\_DT\_new');

EXEC dbms\_data\_mining.drop\_model('Beer\_DT\_new');

COMMENT ON MINING MODEL MLSchema.Beer\_DT IS 'Decision Tree model predicts beer overall rating';

SELECT view\_name, view\_type FROM user\_mining\_model\_views

WHERE model\_name='BEER\_DT'

ORDER BY view\_name;

BEGIN

DBMS\_DATA\_MINING.APPLY(

model\_name => 'Beer\_DT',

data\_table\_name => 'Beer\_testing\_data',

case\_id\_column\_name => 'IDINDEX',

result\_table\_name => 'Beer\_apply\_results\_DT',

data\_schema\_name => 'ML');

END;

/

DBMS\_DATA\_MINING.COMPUTE\_CONFUSION\_MATRIX (

accuracy OUT NUMBER,

apply\_result\_table\_name IN VARCHAR2,

target\_table\_name IN VARCHAR2,

case\_id\_column\_name IN VARCHAR2,

target\_column\_name IN VARCHAR2,

confusion\_matrix\_table\_name IN VARCHAR2,

score\_column\_name IN VARCHAR2 DEFAULT 'PREDICTION',

score\_criterion\_column\_name IN VARCHAR2 DEFAULT 'PROBABILITY',

cost\_matrix\_table\_name IN VARCHAR2 DEFAULT NULL,

apply\_result\_schema\_name IN VARCHAR2 DEFAULT NULL,

target\_schema\_name IN VARCHAR2 DEFAULT NULL,

cost\_matrix\_schema\_name IN VARCHAR2 DEFAULT NULL,

score\_criterion\_type IN VARCHAR2 DEFAULT 'PROBABILITY');

CREATE TABLE Beer\_apply\_results\_DT AS

SELECT IDINDEX,

PREDICTION(Beer\_DT USING \*) prediction,

PREDICTION\_PROBABILITY(Beer\_DT USING \*)probability

FROM Beer\_testing\_data;

DECLARE

v\_accuracy NUMBER;

BEGIN

DBMS\_DATA\_MINING.COMPUTE\_CONFUSION\_MATRIX (

accuracy => v\_accuracy,

apply\_result\_table\_name => 'Beer\_apply\_results\_DT',

target\_table\_name => 'Beer\_testing\_data',

case\_id\_column\_name => 'IDINDEX',

target\_column\_name => 'OVERALL',

confusion\_matrix\_table\_name => 'Beer\_confusion\_matrix\_DT',

score\_column\_name => 'PREDICTION',

score\_criterion\_column\_name => 'PROBABILITY',

cost\_matrix\_table\_name => null,

apply\_result\_schema\_name => null,

target\_schema\_name => null,

cost\_matrix\_schema\_name => null,

score\_criterion\_type => 'PROBABILITY');

END;

/

CREATE TABLE Beer\_costs\_DT (

actual\_target\_value NUMBER,

predicted\_target\_value NUMBER,

cost NUMBER);

INSERT INTO Beer\_costs\_DT values (5, 5, 0);

INSERT INTO Beer\_costs\_DT values (5, 4, .25);

INSERT INTO Beer\_costs\_DT values (5, 3, .50);

INSERT INTO Beer\_costs\_DT values (5, 2, .75);

INSERT INTO Beer\_costs\_DT values (5, 1, 1);

INSERT INTO Beer\_costs\_DT values (4, 5, .25);

…

INSERT INTO Beer\_costs\_DT values (3, 1, .50);

INSERT INTO Beer\_costs\_DT values (2, 5, .75);

INSERT INTO Beer\_costs\_DT values (2, 4, .50);

INSERT INTO Beer\_costs\_DT values (2, 3, .25);

INSERT INTO Beer\_costs\_DT values (2, 2, 0);

INSERT INTO Beer\_costs\_DT values (2, 1, .25);

INSERT INTO Beer\_costs\_DT values (1, 1, 0);

…

COMMIT;

DECLARE

v\_accuracy NUMBER;

BEGIN

DBMS\_DATA\_MINING.COMPUTE\_CONFUSION\_MATRIX (

accuracy => v\_accuracy,

apply\_result\_table\_name => 'Beer\_apply\_results\_DT',

target\_table\_name => 'Beer\_testing\_data',

case\_id\_column\_name => 'IDINDEX',

target\_column\_name => 'OVERALL',

confusion\_matrix\_table\_name => 'Beer\_confusion\_matrix\_DT',

score\_column\_name => 'PREDICTION',

score\_criterion\_column\_name => 'COST',

cost\_matrix\_table\_name => null,

apply\_result\_schema\_name => null,

target\_schema\_name => null,

cost\_matrix\_schema\_name => null,

score\_criterion\_type => 'COST');

END;

/

DECLARE

v\_AUC NUMBER;

BEGIN

DBMS\_DATA\_MINING.COMPUTE\_ROC (

roc\_area\_under\_curve => v\_AUC,

apply\_result\_table\_name => 'Student\_results\_DT',

target\_table\_name => 'Student\_testing\_data',

case\_id\_column\_name => 'Student\_ID',

target\_column\_name => 'Enrolled',

roc\_table\_name => 'Student\_ROC\_DT',

positive\_target\_value => '1',

score\_column\_name => 'PREDICTION',

score\_criterion\_column\_name => 'PROBABILITY');

END;

/

BEGIN

DBMS\_DATA\_MINING.COMPUTE\_LIFT (

apply\_result\_table\_name => 'Student\_results\_DT',

target\_table\_name => 'Student\_testing\_data',

case\_id\_column\_name => 'Student\_ID',

target\_column\_name => 'Enrolled',

lift\_table\_name => 'Student\_Lift\_DT',

positive\_target\_value => '1',

score\_column\_name => 'PREDICTION',

score\_criterion\_column\_name => 'PROBABILITY',

num\_quantiles => 10,

score\_criterion\_type => 'PROBABILITY');

END;

/

EXEC dbms\_data\_mining.apply('Beer\_DT','Beer\_bryggeri', 'IDINDEX', 'Bryggeri\_Result\_Table');

select \* from Bryggeri\_Result\_Table;

SELECT idindex, PREDICTION(Beer\_DT USING \*) pred,

PREDICTION\_DETAILS(Beer\_DT USING \*) preddet

FROM Beer\_bryggeri

WHERE idindex = 6;

SELECT PREDICTION (Beer\_DT USING \*)

FROM beer\_bryggeri where IDINDEX = 6;

SELECT PREDICTION (Beer\_DT USING STYLE)

FROM beer\_bryggeri where IDINDEX = 6;

SELECT PREDICTION(Beer\_DT USING 'Bryggeri' AS text)

FROM DUAL;

SELECT PREDICTION\_PROBABILITY(Beer\_DT, 5 USING \*) as beer\_overall\_prob

FROM beer\_bryggeri

WHERE idindex = 6;

SELECT prediction(STUDENT\_PART\_CLAS\_SVM using \*) pred, ora\_dm\_partition\_name(STUDENT\_PART\_CLAS\_SVM USING \*) partname FROM studentenrolment;

SELECT PREDICTION(/\*+ GROUPING \*/STUDENT\_PART\_CLAS\_SVM USING \*) pred FROM studentenrolment;

select idindex,

CLUSTER\_ID(Beer\_KMEANS USING \*) as clus, CLUSTER\_PROBABILITY(Beer\_KMEANS USING \*) as prob, CLUSTER\_DISTANCE (Beer\_KMEANS USING \*) as dist

from Beer\_bryggeri;

CREATE OR REPLACE DIRECTORY oml\_model\_dir AS '/dm\_path/oml\_models';

GRANT READ, WRITE ON DIRECTORY oml\_model\_dir TO mlusertest;

GRANT READ ON DIRECTORY oml\_model\_dir TO mlusetprod;

BEGIN

dbms\_data\_mining.export\_model (

filename => 'BeerDT.dmp',

directory => 'oml\_model\_dir');

model\_filter => 'name in (''Beer\_DT'')');

END;

/

BEGIN

dbms\_data\_mining.import\_model (

filename => 'BeerDT.dmp',

directory => 'oml\_model\_dir',

schema\_remap => 'TestML:ProdML',

tablespace\_remap => 'EXAMPLE:SYSAUX');

END;

/

DECLARE

v\_blob blob;

BEGIN

dbms\_lob.createtemporary(v\_blob, FALSE);

dbms\_data\_mining.export\_sermodel(v\_blob, 'Beer\_DT');

INSERT INTO modelexpblob (modeldesc) values (v\_blob);

dbms\_lob.freetemporary(v\_blob);

END;

/

DECLARE

v\_blob blob;

BEGIN

-- dbms\_lob.createtemporary(v\_blob, FALSE);

SELECT modeldesc into v\_blob from modelexpblob;

-- you can also fill in v\_blob from a file and then you need

-- the other lines marked as comments

dbms\_data\_mining.import\_sermodel(v\_blob, 'IMP\_MODEL');

-- dbms\_lob.freetemporary(v\_blob);

END;

/

SELECT model\_name, mining\_function, algorithm, algorithm\_type FROM user\_mining\_models WHERE model\_name = 'IMP\_MODEL';

BEGIN

dbms\_data\_mining.import\_model ('PMML\_regression',

XMLType (bfilename ('PMMLDIR', 'HousePrice\_regression.xml'),

nls\_charset\_id ('AL32UTF8')));

END;

/

SELECT student\_id, age, pred\_age, age-pred\_age age\_diff, pred\_det FROM

(SELECT student\_id, age, pred\_age, pred\_det,

RANK() OVER (ORDER BY ABS(age-pred\_age) DESC) rnk FROM

(SELECT student\_id, age,

PREDICTION(FOR age USING \*) OVER () pred\_age,

PREDICTION\_DETAILS(FOR age ABS USING \*) OVER () pred\_det

FROM studentenrolment))

WHERE rnk <= 5;

BEGIN

INSERT INTO Student\_part\_settings VALUES

(dbms\_data\_mining.algo\_name,

dbms\_data\_mining.algo\_support\_vector\_machines);

INSERT INTO Student\_part\_settings VALUES

(dbms\_data\_mining.prep\_auto, dbms\_data\_mining.prep\_auto\_on);

INSERT INTO Student\_part\_settings VALUES

(dbms\_data\_mining.svms\_kernel\_function,dbms\_data\_mining.svms\_linear);

-- define that it will be partitioned by GENDER

INSERT INTO Student\_part\_settings VALUES

(dbms\_data\_mining.odms\_partition\_columns, 'GENDER');

COMMIT;

END;

/

BEGIN

DBMS\_DATA\_MINING.CREATE\_MODEL(

model\_name => 'Student\_part\_SVM',

mining\_function => dbms\_data\_mining.classification,

data\_table\_name => 'Studentenrolment',

case\_id\_column\_name => 'student\_id',

target\_column\_name => 'enrolled',

settings\_table\_name => 'Student\_part\_settings');

END;

/

select MODEL\_NAME as model, PARTITION\_NAME as partition, POSITION, COLUMN\_NAME as cname, COLUMN\_VALUE as cvalue from ALL\_MINING\_MODEL\_PARTITIONS;

SELECT attribute\_name, attribute\_type

FROM user\_mining\_model\_attributes

WHERE model\_name = 'STUDENT\_PART\_SVM'

ORDER BY attribute\_name;

SELECT view\_name, view\_type FROM user\_mining\_model\_views

WHERE model\_name='STUDENT\_PART\_SVM'

ORDER BY view\_name;

CREATE USER dmuser IDENTIFIED BY password

DEFAULT TABLESPACE default\_tablespace

TEMPORARY TABLESPACE temp\_tablespace

QUOTA UNLIMITED on default\_tablespace;

GRANT create mining model TO dmuser;

GRANT create procedure TO dmuser;

GRANT create session TO dmuser;

GRANT create table TO dmuser;

GRANT create sequence TO dmuser;

GRANT create view TO dmuser;

GRANT create job TO dmuser;

GRANT create type TO dmuser;

GRANT create synonym TO dmuser;

GRANT EXECUTE ON ctxsys.ctx\_ddl TO dmuser;