HW1-1

a)

Names	Email Address
Qingyang Zhao	jhun0324@berkeley.edu

Description of Team: Best Group Ever

How did I work?

Comments:

b) I certify that all solutions are entirely in my words and that I have not looked at another student's solutions. I have credited all external sources in this write up.

Qingyang Zhao

HW11-2

a) $\langle f, g\rangle_H = \sum_{m=1}^{M}\sum_{s=1}^{S} \alpha_m \beta_s k(y_m, x_s)$ $\quad f(x) = \sum_{m=1}^{M} \alpha_m k(x, y_m)$ $\quad g(x) = \sum_{s=1}^{S} \beta_s k(x, x_s)$

Symmetry: $\langle f, g\rangle_H = \langle g, f\rangle_H$

$\langle f, g\rangle_H = \sum_{m=1}^{M}\sum_{s=1}^{S} \alpha_m \beta_s k(y_m, x_s)$ $\quad \langle g, f\rangle_H = \sum_{s=1}^{S}\sum_{m=1}^{M} \alpha_m \beta_s k(x_s, y_m)$

Since $k(y_m, x_s) = k(x_s, y_m)$ and M S are all finite numbers.

$\langle f, g\rangle_H = \sum_{m=1}^{M}\sum_{s=1}^{S} \alpha_m \beta_s k(y_m, x_s) = \sum_{s=1}^{S}\sum_{m=1}^{M} \alpha_m \beta_s k(x_s, y_m) = \langle g, f\rangle_H \Rightarrow \langle f, g\rangle = \langle g, f\rangle$

Linearity: ① $\langle af, g\rangle_H = \sum_{m=1}^{M}\sum_{s=1}^{S} a\alpha_m \beta_s k(x_s, y_m) = a\sum_{m=1}^{M}\sum_{s=1}^{S} \alpha_m \beta_s k(x_s, y_m) = a\langle f, g\rangle_H$

$\Downarrow$

$\langle af, g\rangle_H = a\langle f, g\rangle_H$

② $h(x) = \sum_{q=1}^{Q} \gamma_q k(x, z_q)$ $\quad h(x) + f(x) = \sum_{m=1}^{M} \alpha_m k(x, y_m) + \sum_{q=1}^{Q} \gamma_q k(x, z_q)$

$= \sum_{r=1}^{R} \eta_r k(x, t_r)$

where, $R = M+Q$, $t_r = \begin{cases} y_r & \text{when } 1 \le r \le M \\ z_{r-M} & \text{when } M < r \le M+Q \end{cases}$

$\langle h+f, g\rangle_H = \sum_{r=1}^{R}\sum_{s=1}^{S} \eta_r \beta_s k(x_s, t_r)$

$= \sum_{r=1}^{M}\sum_{s=1}^{S} \eta_r \beta_s k(x_s, t_r) + \sum_{r=M+1}^{M+Q}\sum_{s=1}^{S} \eta_r \beta_s k(x_s, t_r)$

$= \sum_{r=1}^{M}\sum_{s=1}^{S} \gamma_r \beta_s k(x_s, y_r) + \sum_{r=M+1}^{M+R}\sum_{s=1}^{S} z_{r-M} \beta_s k(x_s, z_{r-M})$

$= \langle f, g\rangle_H + \langle h, g\rangle_H$

Positive-Definiteness:

$\langle f, f\rangle_H = \sum_{i=1}^{S}\sum_{j=1}^{S} \alpha_i \alpha_j k(y_i, y_j)$

$= [\alpha_1 \cdots \alpha_S] \begin{bmatrix} & K & \end{bmatrix} \begin{bmatrix} \alpha_1 \\ | \\ \alpha_S \end{bmatrix}$

$= \vec{\alpha}^T K \vec{\alpha}$

Since K is positive definite

$\langle f, f\rangle_H = \vec{\alpha}^T K \vec{\alpha} > 0 \quad \vec{\alpha} \neq \vec{0}$

when $\vec{\alpha} = \vec{0}$ $\langle f, f\rangle_H = \vec{\alpha}^T K \vec{\alpha} = 0$

which means $f(x) = 0$

norm of function f:

$\sqrt{\langle f, f\rangle_H} = \sqrt{\vec{\alpha}^T K \vec{\alpha}}$

b) $f(s) = k(x, s)$ $\quad g(s) = k(y, s)$

$\langle f(s), g(s)\rangle = \langle k(x, s), k(y, s)\rangle_H = k(x, y) = \langle k(x, \cdot), k(y, \cdot)\rangle_H$

$\langle k(\cdot, x_i), f\rangle_H = \sum_{m=1}^{M} \alpha_m k(x_i, y_m)$

$= f(x_i)$

c) Let $M = \{m \mid m = \sum_{n=1}^{N} \alpha_n k(x_i, x_n) : \alpha_n \in R\}$. Some g s.t. $\langle h', g\rangle_H = 0$

then $f \in H$ $\quad f = m + g$

$$obj = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \langle k(\cdot, x_i), f\rangle_H) + \lambda \langle f, f\rangle_H$$

$$= \frac{1}{N}\sum_{i=1}^{N} L(y_i, \langle k(\cdot, x_i), m\rangle_H + \langle k(\cdot, x_i), g\rangle_H) + \lambda \langle m+g, m+g\rangle_H$$

$k(\cdot, x_i) \in M$, $\langle m+g, m+g\rangle_H = \langle m, m\rangle_H + \langle g, g\rangle_H$

$$obj = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \langle k(\cdot, x_i), m\rangle_H) + \lambda \langle m, m\rangle_H + \lambda \langle g, g\rangle_H$$

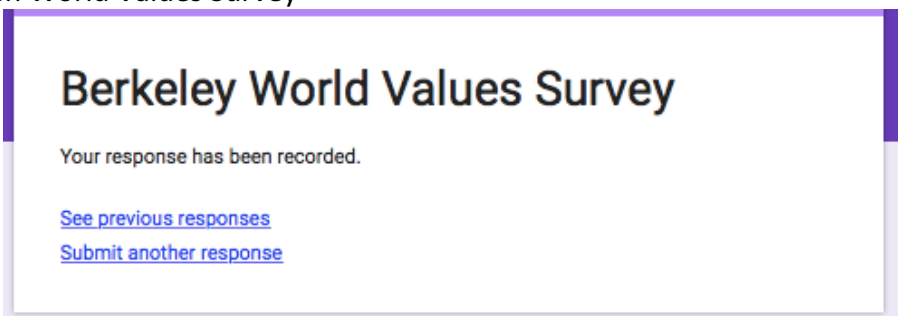
choose the smallest from every m

$$\geq \min_{m \in M} \frac{1}{N}\sum_{i=1}^{N} L(y_i, \langle k(\cdot, x_i), m\rangle_H) + \lambda \langle m, m\rangle_H$$

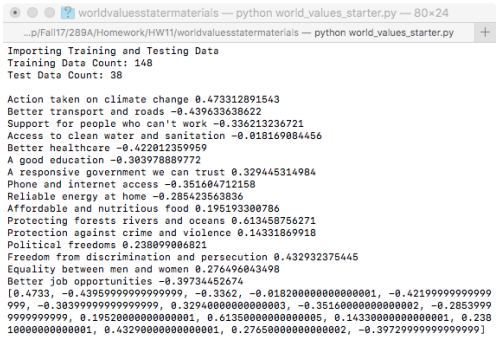
Equality holds when $\langle g, g\rangle_H = 0$, which means $f^* = m$

HW 11 – 3

a) Finish World Values Survey

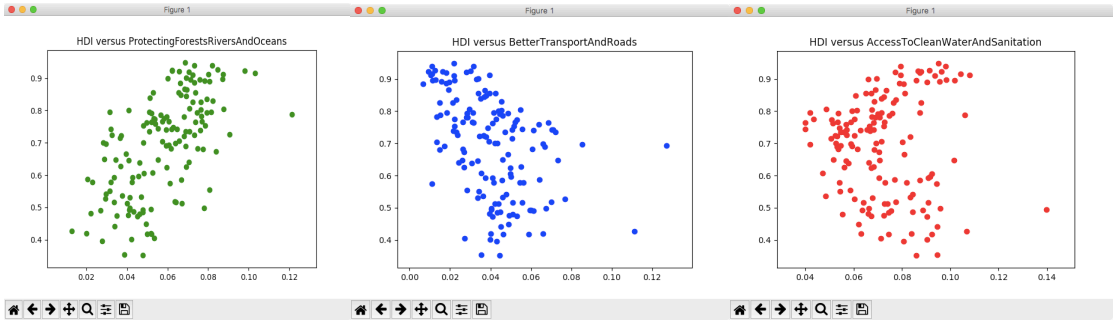


b)

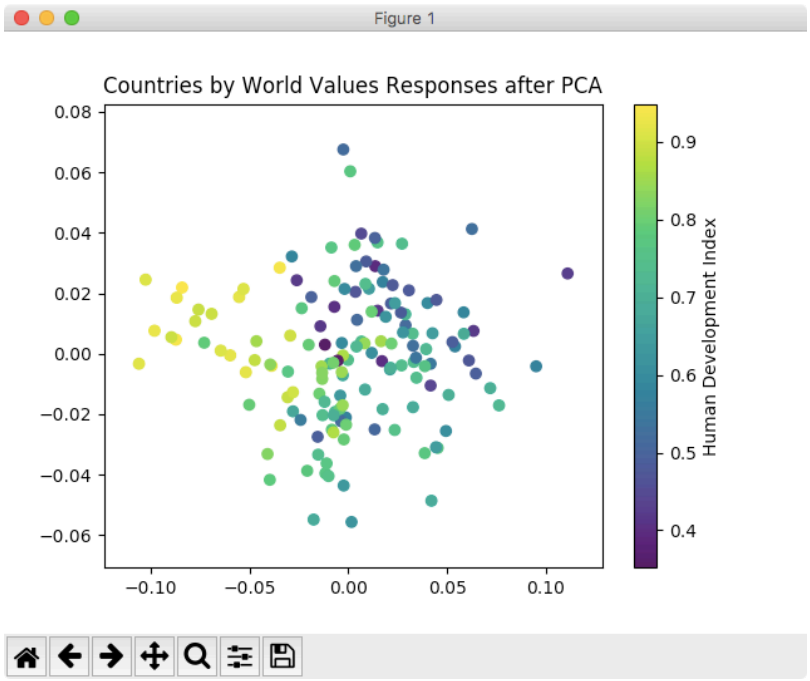


According to the figures above.
Most positive: Protecting forests rivers and oceans (0.6135)
Most Negative: Better Transport and roads (-0.4396)
Least correlated: access to clean water and sanitation (-0.0181)

c)
positive, negative, least correlated:



d)



e) Ridge Regression

```
Ridge Regression
RMSE: 0.12303337350607801
Pipeline(steps=[('ridge', Ridge(alpha=0.02, copy_X=True, fit_intercept=True, max
_iter=None,
   normalize=False, random_state=None, solver='auto', tol=0.001))])
[[ 0.80823467 -0.74985758 -0.17800015 -1.28408103 -0.66293176 -0.82203172
   0.73733884 -0.92891581 -0.82049672  0.39614952  2.0708291  -0.06718981
   0.48310656  0.72671425  0.42921192 -0.13808023]]
```

f) Lasso Regression

```
Lasso Regression
RMSE: 0.12602242808947522
Pipeline(steps=[('lasso', Lasso(alpha=0.00020000000000000001, copy_X=True, fit_i
ntercept=True,
   max_iter=1000, normalize=False, positive=False, precompute=False,
   random_state=None, selection='cyclic', tol=0.0001, warm_start=False))])
[ 0.1590192  -0.72844929 -0.         -0.85945074 -0.66274144 -0.02556703
  0.33904781 -0.29897158 -0.          0.          3.48536375  0.          0.
  0.87057995  0.32897045 -0.        ]
```

g)
Yeah, more zeros.

h)
take the average of the neighbours.

i)

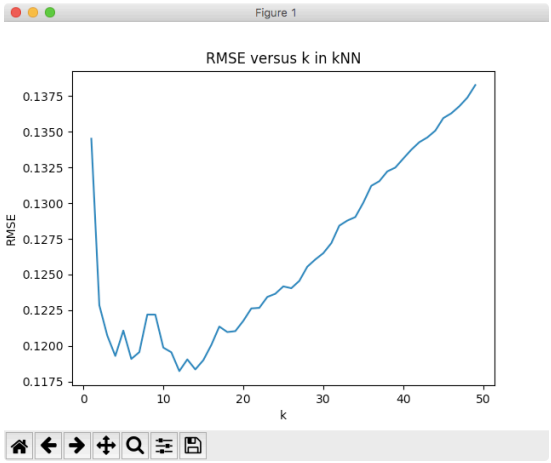
```
          Ireland
United Kingdom
          Belgium
          Finland
            Malta
          Austria
           France
```

j) K nearest neighbor, k = 12

```
k Nearest Neighbors Regression
RMSE: 0.1182458946077689
Pipeline(steps=[('knn', KNeighborsRegressor(algorithm='auto', leaf_size=30, metr
ic='minkowski',
          metric_params=None, n_jobs=1, n_neighbors=12, p=2,
          weights='uniform'))])
```

k)

l) best is 12 or 14



m) after scaling

70	Sweden
5	Netherlands
110	France
102	Switzerland
39	Germany
108	Finland
24	Norway

n) k = 3
RMSE = 0.11567

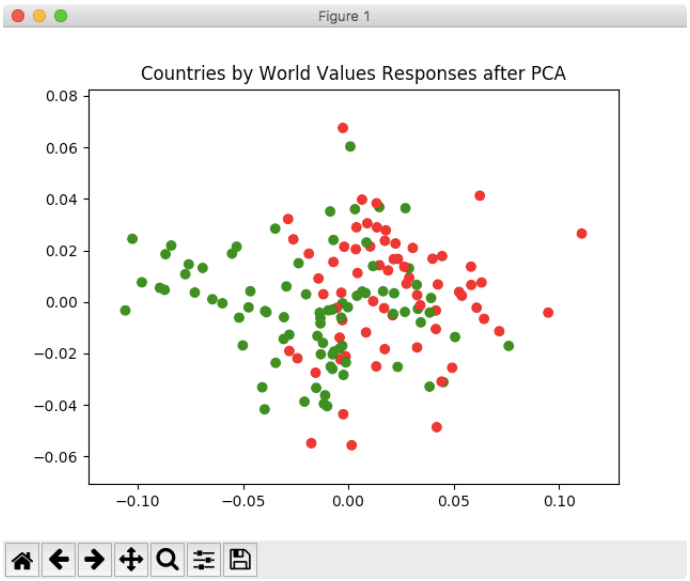
```
k Nearest Neighbors Regression
RMSE: 0.11567129764595492
Pipeline(steps=[('scale', StandardScaler(copy=True, with_mean=True, with_std=Tru
e)), ('knn', KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='minkows
ki',
          metric_params=None, n_jobs=1, n_neighbors=3, p=2,
          weights='uniform'))])
```

o)

p) prediction:

```
[[-0.1804483 ]
 [-0.06741022]
 [-0.00338615]
 [-0.01141103]
 [ 0.08166901]
 [-0.09323173]
 [-0.0097831 ]
 [-0.01562966]
 [ 0.08596107]
 [ 0.1138649 ]
 [-0.01878029]
 [-0.00675476]
 [ 0.06445697]
 [ 0.01351135]
 [-0.01379985]
 [-0.15267183]
 [ 0.00610526]
 [ 0.13723134]
 [ 0.01299407]
 [ 0.08321071]
 [-0.05823427]
 [-0.04194346]
 [-0.29624551]
 [ 0.0432918 ]
 [-0.07548227]
 [ 0.07955382]
 [-0.25385226]
 [ 0.21182503]
 [-0.08436168]
 [-0.05257132]
 [ 0.08113328]
 [ 0.17239041]
 [ 0.03667984]
 [ 0.2174793 ]
 [ 0.22206283]
 [ 0.15102156]
 [-0.17457403]
 [ 0.09718166]]
```

q)
r) two classes PCA



s) not good this is not linearly separable
t) 0.75

```
SVM Classification
Accuracy: 0.75
Pipeline(steps=[('svm', SVC(C=48.0, cache_size=200, class_weight=None, coef0=0.0
,
  decision_function_shape=None, degree=3, gamma='auto', kernel='linear',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False))])
```

u)

0.743 after pca and scaling

```
SVM Classification
Accuracy: 0.743243243243
Pipeline(steps=[('pca', PCA(copy=True, iterated_power='auto', n_components=None, random_state=None,
  svd_solver='auto', tol=0.0, whiten=False)), ('scale', StandardScaler(copy=True, with_mean=True, with_std=True)), ('svm', S
VC(C=0.069999999999999993, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape=None, degree=3, gamma='auto', kernel='linear',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False))])
```

v) after rbf : 0.804

```
SVM Classification
Accuracy: 0.804054054054
Pipeline(steps=[('pca', PCA(copy=True, iterated_power='auto', n_components=None, random_state=None,
  svd_solver='auto', tol=0.0, whiten=False)), ('scale', StandardScaler(copy=True, with_mean=True, with_std=True)), ('svm', S
VC(C=0.48999999999999999, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape=None, degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False))])
```

w)

KNN:

```
k Nearest Neighbors Classification
Accuracy: 0.763513513514
Pipeline(steps=[('knn', KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=1, n_neighbors=4, p=2,
           weights='distance'))])
```

After scaling:

```
k Nearest Neighbors Classification
Accuracy: 0.763513513514
Pipeline(steps=[('knn', KNeighborsClassifier(algorithm='auto', leaf_size=30, met
ric='minkowski',
           metric_params=None, n_jobs=1, n_neighbors=4, p=2,
           weights='distance'))])
```

x)

y)

z)