Lab 2 Solutions

(g)

```
# Correction Matrix Plot (generic)
from matplotlib import pyplot
from pandas import read_csv
filename = 'pima-indians-diabetes.data.csv'
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
data = read_csv(filename, names=names)
correlations = data.corr()
# plot correlation matrix
fig = pyplot.figure()
ax = fig.add_subplot(111)
cax = ax.matshow(correlations, vmin=-1, vmax=1)
fig.colorbar(cax)
pyplot.show()
```

Generating the plot, you can see that it gives the same information although making it a little harder to see what attributes are correlated by name. Use this generic plot as a first cut to understand the correlations in your dataset and customise it like the first example in order to read off more specific data if needed.

(k)

```
dataframe.plot(kind='density', subplots=True, layout=(3,3), sharex=False, figsize=[20, 20])
pyplot.show()
```

(l)

```
import seaborn as sns
sns.distplot(rescaledX[:,1])
sns.distplot(rescaledX[:,2])
sns.distplot(rescaledX[:,3])
sns.distplot(rescaledX[:,4])
sns.distplot(rescaledX[:,5])
sns.distplot(rescaledX[:,6])
sns.distplot(rescaledX[:,7])
```

(o)

```
# Normalize data (length of 1)
from sklearn.preprocessing import Normalizer
from pandas import read_csv
from numpy import set_printoptions
filename = 'pima-indians-diabetes.data.csv'
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
dataframe = read_csv(filename, names=names)
array = dataframe.values
```

```
# separate array into input and output components
X = array[:,0:8]
Y = array[:,8]
scaler = Normalizer().fit(X)
normalizedX = scaler.transform(X)

# summarize transformed data
set_printoptions(precision=3)
print("Normalised Samples \n", normalizedX[0:5,:])

# Decision tree classification
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.tree import DecisionTreeClassifier
kfold = KFold(n_splits=10, random_state=7)
model = DecisionTreeClassifier()
results1 = cross_val_score(model, X, Y, cv=kfold)
print("Mean estimated accuracy \n",results1.mean())

# Decision tree classification on normalised data
results2 = cross_val_score(model, normalizedX, Y, cv=kfold)
print("Mean estimated accuracy on normalised data \n",results2.mean())
```