Problem Statement

The purpose is to predict whether the Pima Indian women shows signs of diabetes or not. We are using a dataset collected by "National Institute of Diabetes and Digestive and Kidney Diseases" which consists of a number of attributes which would help us to perform this prediction.

Constraints on data collection All patients whose data has been collected are females at least 21 years old of Pima Indian heritage

Import all the Libraries

In [52]:

```
import pandas as pd
import numpy as np
import warnings
import random
import matplotlib.pyplot as plt
import seaborn as sns
numbins = 10
```

In [53]:

```
data = pd.read_csv(r'C:\Users\Anas Khanooni\Desktop\project\pima-indians-diabetes.csv')
```

In [54]:

```
colnames = ['preg','plas','pres','skin','test','mass','pedi','age','class']
data = pd.read_csv('pima-indians-diabetes.csv')
```

In [55]:

```
data.head(10)
```

Out[55]:

	Preg	Plas	Pres	skin	test	mass	pedi	age	class
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
5	5	116	74	0	0	25.6	0.201	30	0
6	3	78	50	32	88	31.0	0.248	26	1
7	10	115	0	0	0	35.3	0.134	29	0
8	2	197	70	45	543	30.5	0.158	53	1
9	8	125	96	0	0	0.0	0.232	54	1

3. Print the datatypes of each column and the shape of the dataset

In [56]:

```
data.shape
```

Out[56]:

```
(768, 9)
```

In [57]:

```
data.dtypes
```

Out[57]:

```
Preg       int64
Plas       int64
Pres       int64
skin       int64
test       int64
mass     float64
pedi     float64
age        int64
class      int64
dtype: object
```

4. Replace all the 0s in the columns with the median of the same column value accordingly.

In [58]:

```
data.loc[data.Plas ==0,'Plas'] = data.Plas.median()
data.loc[data.Pres ==0,'Pres'] = data.Pres.median()
data.loc[data.skin ==0,'skin'] = data.skin.median()
data.loc[data.test ==0,'test'] = data.test.median()
data.loc[data.mass ==0,'mass'] = data.mass.median()
```

5. Print the descriptive statistics of each & every column using describe() function

In [59]:

```
data.describe().transpose ()
```

Out[59]:

	count	mean	std	min	25%	50%	75%	max
Preg	768.0	3.845052	3.369578	0.000	1.00000	3.0000	6.00000	17.00
Plas	768.0	121.656250	30.438286	44.000	99.75000	117.0000	140.25000	199.00
Pres	768.0	72.386719	12.096642	24.000	64.00000	72.0000	80.00000	122.00
skin	768.0	27.334635	9.229014	7.000	23.00000	23.0000	32.00000	99.00
test	768.0	94.652344	105.547598	14.000	30.50000	31.2500	127.25000	846.00
mass	768.0	32.450911	6.875366	18.200	27.50000	32.0000	36.60000	67.10
pedi	768.0	0.471876	0.331329	0.078	0.24375	0.3725	0.62625	2.42
age	768.0	33.240885	11.760232	21.000	24.00000	29.0000	41.00000	81.00
class	768.0	0.348958	0.476951	0.000	0.00000	0.0000	1.00000	1.00

6. See the distribution of 'Class' variable and plot it using approriate

In [60]:

```
data.groupby('class').agg({'class': 'count'})
```

Out[60]:

	class
class	
0	500
1	268

7. Use pairplots and correlation method to observe the relationship between different variables and state your insights.

In [61]:

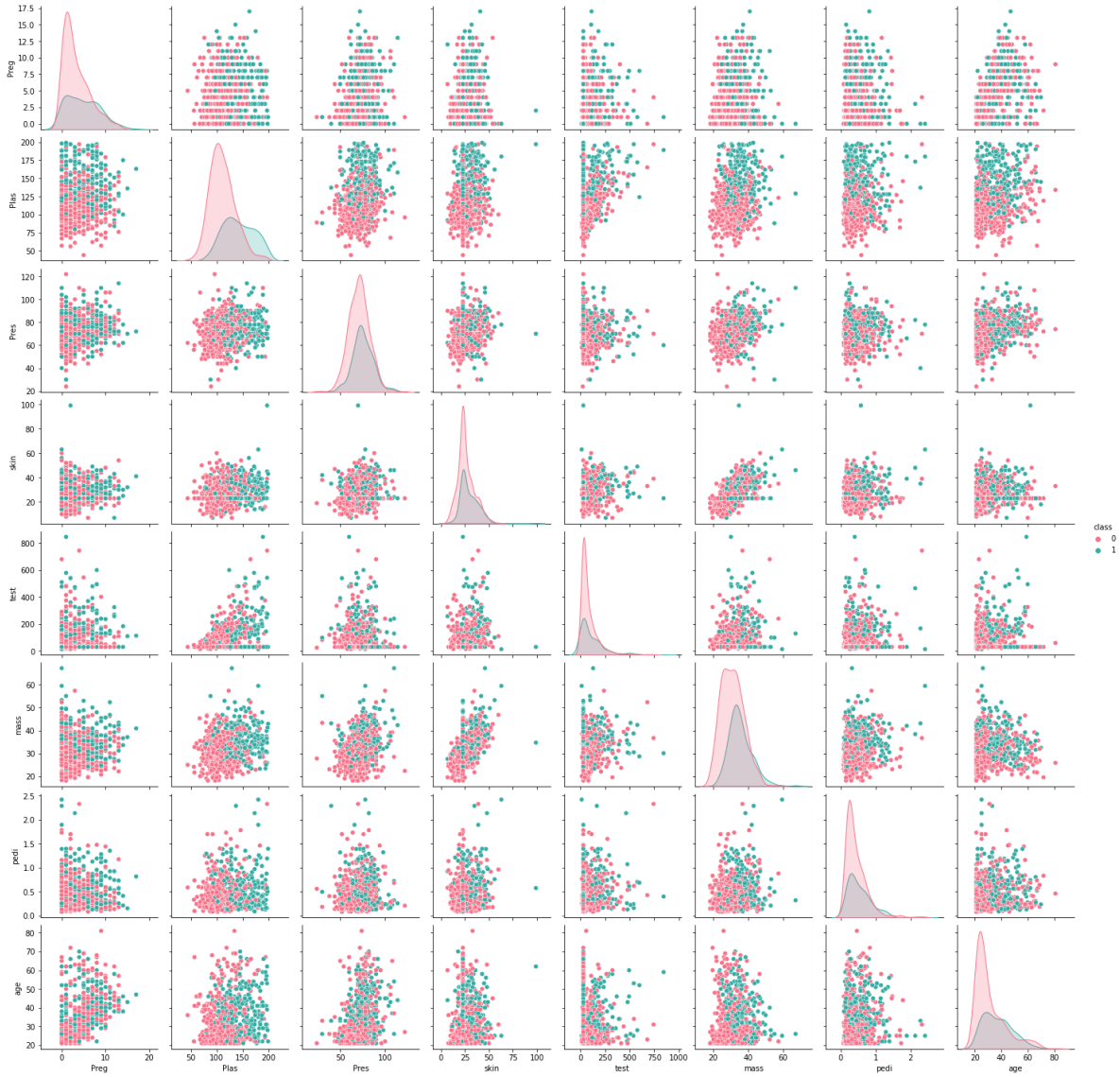
```
# Hint: Use seaborn plot and check the relationship between different variables
```

In [62]:

```
import seaborn as sns
sns.pairplot(data, hue='class', palette='husl')
```

Out[62]:

```
<seaborn.axisgrid.PairGrid at 0x1c3d22ed280>
```



In [63]:

```
data.corr()
```

Out[63]:

	Preg	Plas	Pres	skin	test	mass	pedi	age	c
Preg	1.000000	0.128213	0.208615	0.032568	-0.055697	0.021546	-0.033523	0.544341	0.22
Plas	0.128213	1.000000	0.218937	0.172143	0.357573	0.231400	0.137327	0.266909	0.49
Pres	0.208615	0.218937	1.000000	0.147809	-0.028721	0.281132	-0.002378	0.324915	0.16
skin	0.032568	0.172143	0.147809	1.000000	0.238188	0.546951	0.142977	0.054514	0.18
test	-0.055697	0.357573	-0.028721	0.238188	1.000000	0.189022	0.178029	-0.015413	0.14
mass	0.021546	0.231400	0.281132	0.546951	0.189022	1.000000	0.153506	0.025744	0.31
pedi	-0.033523	0.137327	-0.002378	0.142977	0.178029	0.153506	1.000000	0.033561	0.17
age	0.544341	0.266909	0.324915	0.054514	-0.015413	0.025744	0.033561	1.000000	0.23
class	0.221898	0.492782	0.165723	0.189065	0.148457	0.312249	0.173844	0.238356	1.00

Check for the correlation between variables whose values are >0.8

Observations:

Diagonal plots have already been discussed in the Observations I of Univariate Analysis. There are no linear relationships between any two variables. There is no strong correlation between any independent variable and class variable.

Using the plot - infer the relationship between different variables.5

8. Split the data into training and test set in the ratio of 70:30(Training:Test).

In [64]:

```
# Splitting data into training and test set for independent attributes
n = data['class'].count()
train_set = data.head(int(round(n*0.7))) #Up to the last initial training set row
test_set = data.tail(int(round(n*0.3))) # Past the last initial training set row

#Capture the target column ('class') into seperate vectors for training set and test set
train_labels = train_set.pop('class')
test_labels = test_set.pop('class')
```

9. Create the decision tree model using "entropy" method of reducing the entropy and fit the it to training data.

In [65]:

```
from sklearn.tree import DecisionTreeClassifier
dt_model = DecisionTreeClassifier(criterion = 'entropy')
dt_model.fit(train_set, train_labels)
```

Out[65]:

```
DecisionTreeClassifier(criterion='entropy')
```

10. Print the accuracy of the model & print confusion matrix

In [66]:

```
dt_model.score(test_set, test_labels)
test_pred = dt_model.predict(test_set)
```

In [67]:

```
print (pd.DataFrame(dt_model.feature_importances_, columns = ["Imp"], index = train_set.col
```

```
           Imp
Preg  0.060162
Plas  0.306329
Pres  0.076010
skin  0.047292
test  0.064692
mass  0.158466
pedi  0.183838
age   0.103211
```

In [ ]: