In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [3]:
```
data = pd.read_csv("C:/Users/Admin-PC/Downlads/pima-data.csv")
```

In [6]:
```
data.shape
```

Out[6]:
```
(768, 10)
```

In [7]:
```
data.describe()
```

Out[7]:

	num_preg	glucose_conc	diastolic_bp	thickness	insulin	bmi	diab_pred	age	skin
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.809136
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.628517
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.906200
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.260800
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	3.900600

In [5]:
```
data.head(10)
```

Out[5]:

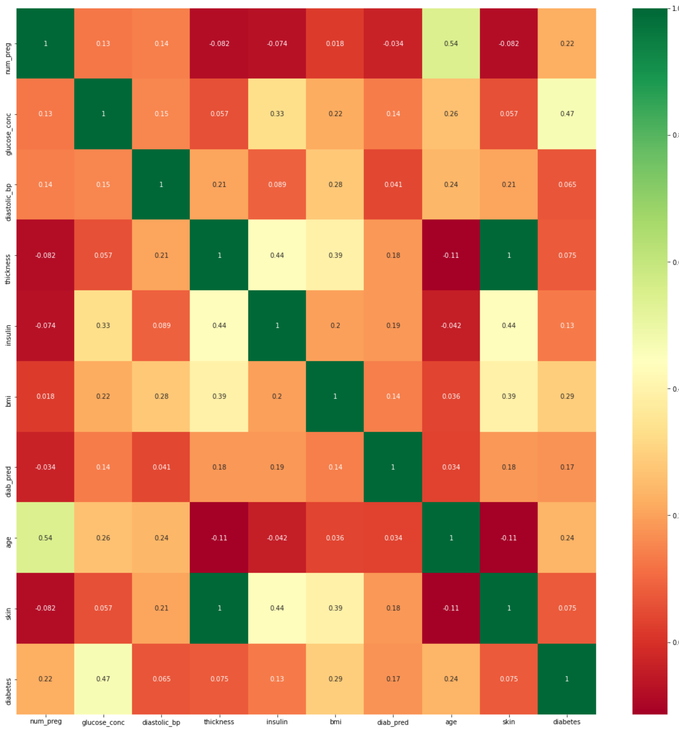
	num_preg	glucose_conc	diastolic_bp	thickness	insulin	bmi	diab_pred	age	skin	diabetes
0	6	148	72	35	0	33.6	0.627	50	1.3790	True
1	1	85	66	29	0	26.6	0.351	31	1.1426	False
2	8	183	64	0	0	23.3	0.672	32	0.0000	True
3	1	89	66	23	94	28.1	0.167	21	0.9062	False
4	0	137	40	35	168	43.1	2.288	33	1.3790	True
5	5	116	74	0	0	25.6	0.201	30	0.0000	False
6	3	78	50	32	88	31.0	0.248	26	1.2608	True
7	10	115	0	0	0	35.3	0.134	29	0.0000	False
8	2	197	70	45	543	30.5	0.158	53	1.7730	True
9	8	125	96	0	0	0.0	0.232	54	0.0000	True

In [8]:
```
data.isnull().values.any()
```

Out[8]:
```
False
```

In [9]:
```
import seaborn as sns
import matplotlib.pyplot as plt
#get correlations of each features in dataset
corrmat = data.corr()
top_corr_features = corrmat.index
plt.figure(figsize=(20,20))
#plot heat map
g=sns.heatmap(data[top_corr_features].corr(),annot=True,cmap="RdYlGn")
```

Out[9]:



In [10]:
```
data.corr()
```

Out[10]:

	num_preg	glucose_conc	diastolic_bp	thickness	insulin	bmi	diab_pred	age	skin	diabetes
num_preg	1.000000	0.129459	0.141282	-0.081672	-0.073535	0.017683	-0.033523	0.544341	-0.081672	0.221898
glucose_conc	0.129459	1.000000	0.152590	0.057328	0.331357	0.221071	0.137337	0.263514	0.057328	0.466581
diastolic_bp	0.141282	0.152590	1.000000	0.207371	0.088933	0.281805	0.041265	0.239528	0.207371	0.065068
thickness	-0.081672	0.057328	0.207371	1.000000	0.436783	0.392573	0.183928	-0.113970	1.000000	0.074752
insulin	-0.073535	0.331357	0.088933	0.436783	1.000000	0.197859	0.185071	-0.042163	0.436783	0.130548
bmi	0.017683	0.221071	0.281805	0.392573	0.197859	1.000000	0.140647	0.036242	0.392573	0.292695
diab_pred	-0.033523	0.137337	0.041265	0.183928	0.185071	0.140647	1.000000	0.033561	0.183928	0.173844
age	0.544341	0.263514	0.239528	-0.113970	-0.042163	0.036242	0.033561	1.000000	-0.113970	0.238356
skin	-0.081672	0.057328	0.207371	1.000000	0.436783	0.392573	0.183928	-0.113970	1.000000	0.074752
diabetes	0.221898	0.466581	0.065068	0.074752	0.130548	0.292695	0.173844	0.238356	0.074752	1.000000

In [11]:
```
diabetes_map = {True: 1, False: 0}
```

In [12]:
```
data['diabetes'] = data['diabetes'].map(diabetes_map)
```

In [14]:
```
data.head(10)
```

Out[14]:

	num_preg	glucose_conc	diastolic_bp	thickness	insulin	bmi	diab_pred	age	skin	diabetes
0	6	148	72	35	0	33.6	0.627	50	1.3790	1
1	1	85	66	29	0	26.6	0.351	31	1.1426	0
2	8	183	64	0	0	23.3	0.672	32	0.0000	1
3	1	89	66	23	94	28.1	0.167	21	0.9062	0
4	0	137	40	35	168	43.1	2.288	33	1.3790	1
5	5	116	74	0	0	25.6	0.201	30	0.0000	0
6	3	78	50	32	88	31.0	0.248	26	1.2608	1
7	10	115	0	0	0	35.3	0.134	29	0.0000	0
8	2	197	70	45	543	30.5	0.158	53	1.7730	1
9	8	125	96	0	0	0.0	0.232	54	0.0000	1

In [15]:
```
diabetes_true_count = len(data.loc[data['diabetes'] == True])
diabetes_false_count = len(data.loc[data['diabetes'] == False])
```

In [16]:
```
(diabetes_true_count,diabetes_false_count)
```

Out[16]:
```
(268, 500)
```

In [17]:
```
from sklearn.model_selection import train_test_split
feature_columns = ['num_preg', 'glucose_conc', 'diastolic_bp', 'insulin', 'bmi', 'diab_pred', 'age', 'skin']
predicted_class = ['diabetes']
```

In [18]:
```
X = data[feature_columns].values
y = data[predicted_class].values

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.30, random_state=10)
```

In [19]:
```
print("total number of rows : {0}".format(len(data)))
print("number of rows missing glucose_conc: {0}".format(len(data.loc[data['glucose_conc'] == 0])))
print("number of rows missing diastolic_bp: {0}".format(len(data.loc[data['diastolic_bp'] == 0])))
print("number of rows missing insulin: {0}".format(len(data.loc[data['insulin'] == 0])))
print("number of rows missing bmi: {0}".format(len(data.loc[data['bmi'] == 0])))
print("number of rows missing diab_pred: {0}".format(len(data.loc[data['diab_pred'] == 0])))
print("number of rows missing age: {0}".format(len(data.loc[data['age'] == 0])))
print("number of rows missing skin: {0}".format(len(data.loc[data['skin'] == 0])))
```

```
total number of rows : 768
number of rows missing glucose_conc: 5
number of rows missing glucose_conc: 5
number of rows missing diastolic_bp: 35
number of rows missing insulin: 374
number of rows missing bmi: 11
number of rows missing diab_pred: 0
number of rows missing age: 0
number of rows missing skin: 227
```

In [ ]:
```

```

In [20]:
```
from sklearn.ensemble import RandomForestClassifier

random_forest_model = RandomForestClassifier(random_state=10)
random_forest_model.fit(X_train, y_train.ravel())
```

Out[20]:
```
RandomForestClassifier(random_state=10)
```

In [21]:
```
predict_train_data = random_forest_model.predict(X_test)

from sklearn import metrics

print("Accuracy = {0:.3f}".format(metrics.accuracy_score(y_test, predict_train_data)))
```

```
Accuracy = 0.749
```

In [ ]:
```

```