INTELLIGENT SYSTEMS LAB-2 (23/08/2021)

__NAME__: Harshvardhan Agarwal | __SEC__: A | __REG NO__.: 201800524

SOURCE CODE

For linear regression:

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.linear_model import LinearRegression
```

In [2]:
```
data = pd.read_csv('Poly_Dataset.csv')
```

In [15]:
```
X = data.iloc[:, 0:1].values
y = data.iloc[:, 1].values
```

In [11]:
```
from sklearn.model_selection import train_test_split
```

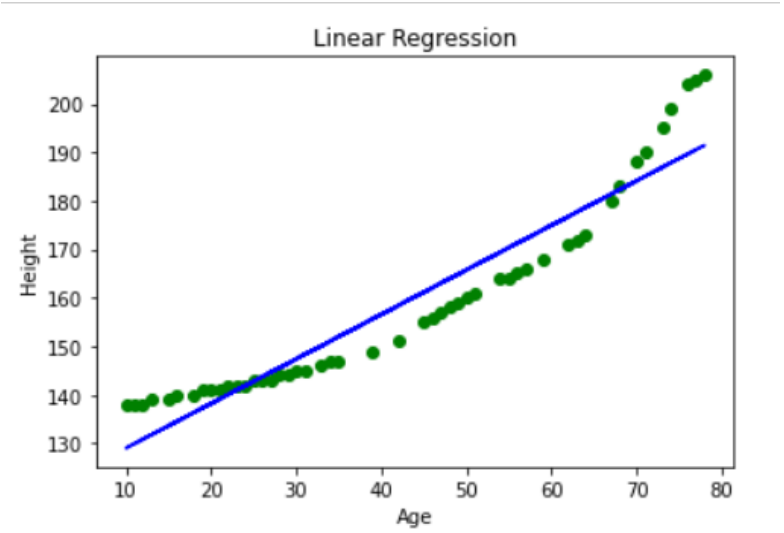
In [16]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
```

In [17]:
```
LinReg = LinearRegression()
LinReg.fit(X_train, y_train)
```

Out[17]:
```
LinearRegression()
```

In [18]:
```
plt.scatter(X_train, y_train, color = 'green')

plt.plot(X_train, LinReg.predict(X_train), color = 'blue')
plt.title('Linear Regression')
plt.xlabel('Age')
plt.ylabel('Height')

plt.show()
```

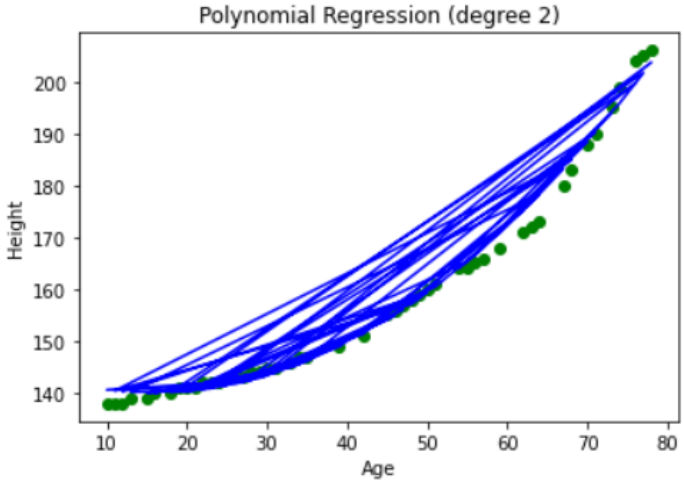
i) Graph for Linear Regression

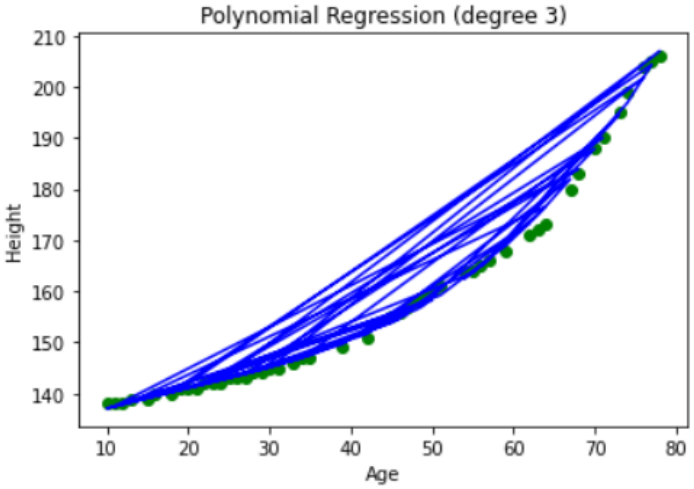


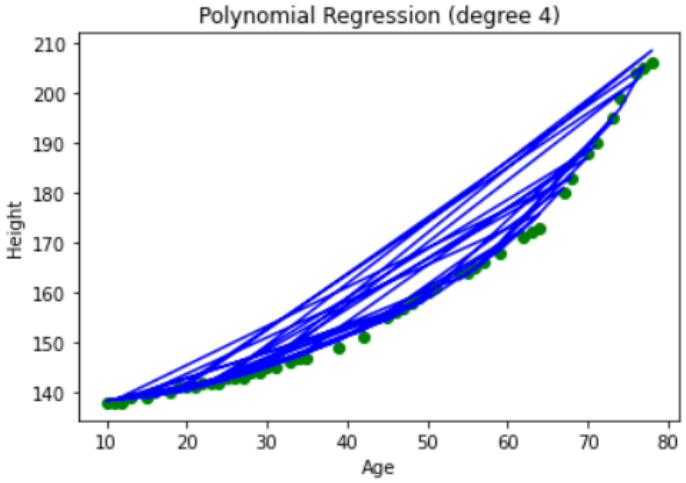
Code for Polynomial Regression

In [19]:
```
##Polynomial regression
from sklearn.preprocessing import PolynomialFeatures

polynom = PolynomialFeatures(degree = 2)
#modify degree for 3 and 4
X_polynom = polynom.fit_transform(X_train)
PolyReg = LinearRegression()
PolyReg.fit(X_polynom, y_train)

plt.scatter(X_train, y_train, color = 'green')
plt.plot(X_train, PolyReg.predict(polynom.fit_transform(X_train)), color = 'blue')
plt.title('Polynomial Regression')
plt.xlabel('Age')
plt.ylabel('Height')
plt.show()

y_predict_slr = LinReg.predict(X_test)
```

ii) Graph for Polynomial Regression (degree 2, 3, 4)







iii) R-squared evaluation for Linear Regression

Add the following lines of code:

```
In [23]: #Model Evaluation using R-Square for Simple Linear Regression
         from sklearn import metrics
         r_square = metrics.r2_score(y_test, y_predict_slr)
         print('R-Square Error associated with Simple Linear Regression:', r_square)

         R-Square Error associated with Simple Linear Regression: 0.8727873738671587
```

R-squared evaluation for Polynomial Regression (degree 2,3,4)

Add the following lines of code:

```
In [26]: y_predict_pr = PolyReg.predict(polynom.fit_transform(X_test))

         #Model Evaluation using R-Square for Polynomial Regression
         from sklearn import metrics
         r_square = metrics.r2_score(y_test, y_predict_pr)
         print('R-Square Error associated with Polynomial Regression is:', r_square)
```

R-Square Error associated with Polynomial Regression (degree 2) is:
0.9892511710983519

R-Square Error associated with Polynomial Regression (degree 3) is:
0.9927634092096052

R-Square Error associated with Polynomial Regression (degree 4) is:
0.9888925536600215

iv) Prediction for Linear Regression

Add the following lines of code:

```
In [32]: Lin_Pred = LinReg.predict([[31]])
         Lin_Pred

Out[32]: array([148.3031158])
```

Prediction for Polynomial Regression (degree 2, 3, 4)

Let's check prediction for the same value in different degrees

Add the following lines of code:

```
In [34]: PolyReg.predict(polynom.fit_transform([[53]]))
```

For degree 2, height predicted for age 53: 162.81769144

For degree 3, height predicted for age 53: 161.33273189

For degree 4, height predicted for age 53: 161.58598092