**Modelling and Analysis Using Non-Linear Regression**

**Abstract:**

This project explores the application of various Non-Linear regression models. We implemented several nonlinear models alongside polynomial regression, including exponential, power, and saturation growth models. The models were evaluated using key performance metrics such as Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and the coefficient of determination . While the nonlinear models provided moderate fits, the polynomial regression model of degree 3 exhibited the most accurate predictions. Visualizations were generated to compare the performance of each model. We use the random data to predict the relationship between dependent and independent variables.

**Introduction:**

Regression analysis involves identifying the relationship between a dependent variable and one or more independent variables. It is one of the most important statistical tools which is extensively used in almost all sciences. It is specially used in business and economics to study the relationship between two or more variables that are related causally. A model of the relationship is hypothesized, and estimates of the parameter values are used to develop an estimated regression equation. Various tests are then employed to determine if the model is satisfactory. Model validation is an important step in the modelling process and helps assess models' reliability before they can be used in decision-making.

**Objective:**

The primary objective of this project is to implement and compare different regression models (nonlinear and polynomial) to predict and evaluate the performance of these models using standard error metrics such as RMSE, MAE, and .

#### ****Exploratory Data Analysis:****

To begin with, we visualized the relationship between dependent variable and Independent variable using a scatter plot. This allowed us to gain insight into the data distribution and potential patterns that could be captured through regression models.

* **Scatter Plot**: The scatter plot revealed that there is a positive correlation between dependent variable and Independent variable, with a tendency for the data points to curve upwards. This suggests that a nonlinear model might be appropriate for fitting the data.

**Regression Models:**

To model the relationship between dependent variable and Independent variable, we apply nonlinear regression models.

To get the non-linear model we need to transform our data into linear model i.e., in the form

The different nonlinear models were tested:

* **Exponential Model**:

Given, best fit to the data. In this model, the constants of the regression model are and . An example of the model is the case of a radioactive dye such as Technetium-99 given to patients who are going through a CT scan of their body to diagnose an illness such as excess fat on a liver. The radiation intensity of the radioactive dye decreases with time and is governed by an exponentially decaying model. The interest in such a model would be to find out how long substantial radiation stays in the body.

Transformation of data through code

#Exponential model

#random data

x = [0, 1, 3, 5, 7, 9]

y = [1, 0.891, 0.708, 0.562, 0.447, 0.355]

z = np.log(y)

Sx = np.sum(x)

Sz = np.sum(z)

xz = x\*z

Sxz = np.sum(xz)

x2 = np.pow(x,2)

Sx2 = np.sum(x2)

a1 = ((n\*Sxz)-(Sz\*Sx))/((n\*Sx2)-(Sx)\*\*2)

a0 = np.mean(z)-a1\*np.mean(x)

a = np.exp(a0)

b = a1

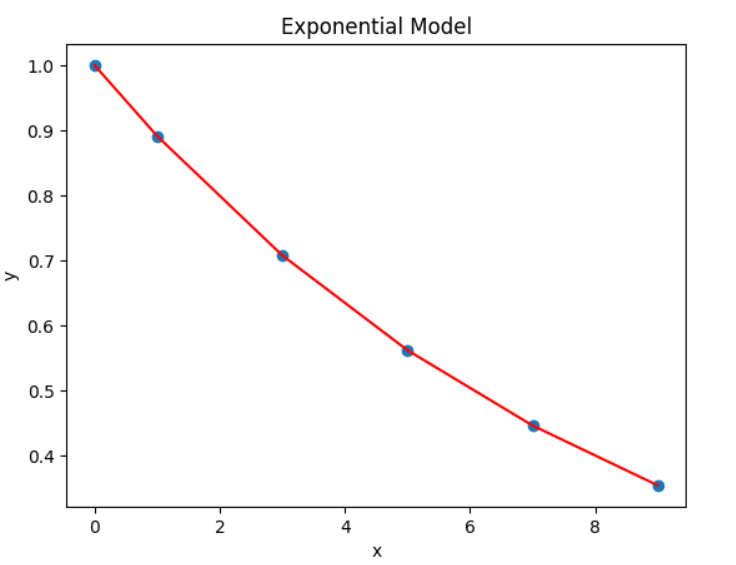
# predected value

pred = []

for i in range(n):

    values = a\*np.exp(b\*t[i])

    pred.append(values)



This is the graph of Exponential model to predict petal length from sepal length

We get the value as 0.999

 **Power Model:**

Given ,best fit to the data. In this model, the constants of the regression model are and . An example of the model is a falling parachutist, where the drag force on the parachute is related through the power of the velocity with which they are falling. An interest in such a model would arise from designing the parachute such that the drag is low enough to maintain control of fall but high enough for the fall to be safe.

Transformation of data through code

#Power model

p = [10, 16, 25, 40, 60]

f = [94, 118, 147, 180, 230]

z = np.log(f)

Sz = np.sum(z)

w = np.log(p)

Sw = np.sum(w)

wz = w\*z

Swz = np.sum(wz)

w2 = np.pow(w,2)

Sw2 = np.sum(w2)

a1 = ((n\*Swz)-(Sz\*Sw))/((n\*Sw2)-(Sw)\*\*2)

a0 = np.mean(z)-a1\*np.mean(w)

a = np.exp(a0)

b = a1

# predected value

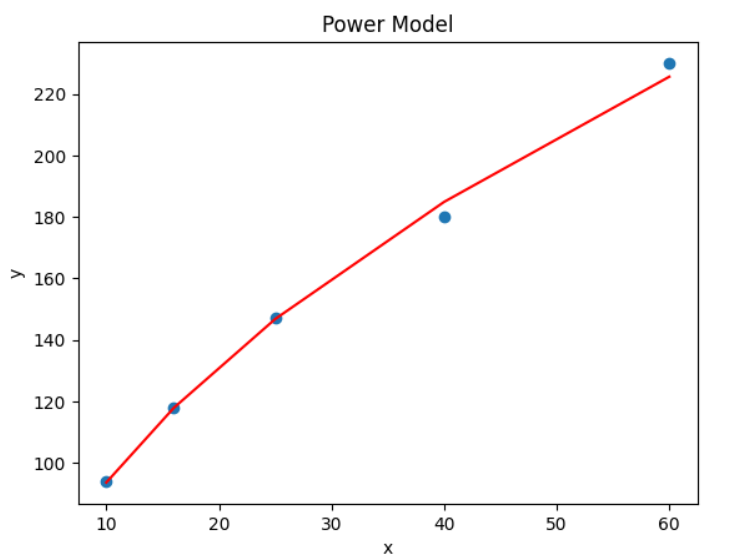
pred = []

for i in range(n):

    values = a\*np.pow(p[i],b)

    pred.append(values)

print(pred)



This is the graph of Power Model to predict petal length from sepal length

We get the value as 0.996

 **Saturation Growth Model**:

Given ,, best fit to the data. In this model, the constants of the regression model are and . An example of the model is the case of how good animation looks. How good an animation looks is measured by a variable called performance and is a function of the frame rate. Frame rate is the frequency at which consecutive frames (images) appear on display. The higher the frame rate, the more natural animation looks to the human eye, but the human eye cannot distinguish the increased performance after a certain frame rate. This model is an example of a saturation growth model where at zero frame rate, the performance is zero as one is looking at a static image, and at high frame rates, the performance level saturates.

Transformation of data through code

#Saturation growth model

p = [10, 16, 25, 40, 60]

f = [94, 118, 147, 180, 230]

z = 1/np.array(f)

w = 1/np.array(p)

Sz = np.sum(z)

Sw = np.sum(w)

wz = w\*z

Swz = np.sum(wz)

w2 = np.pow(w,2)

Sw2 = np.sum(w2)

print(Sw, Sz, Swz, Sw2)

a1 = ((n\*Swz)-(Sz\*Sw))/((n\*Sw2)-(Sw)\*\*2)

a0 = np.mean(z)-a1\*np.mean(w)

a = 1/a0

b = a1\*a

# predected value

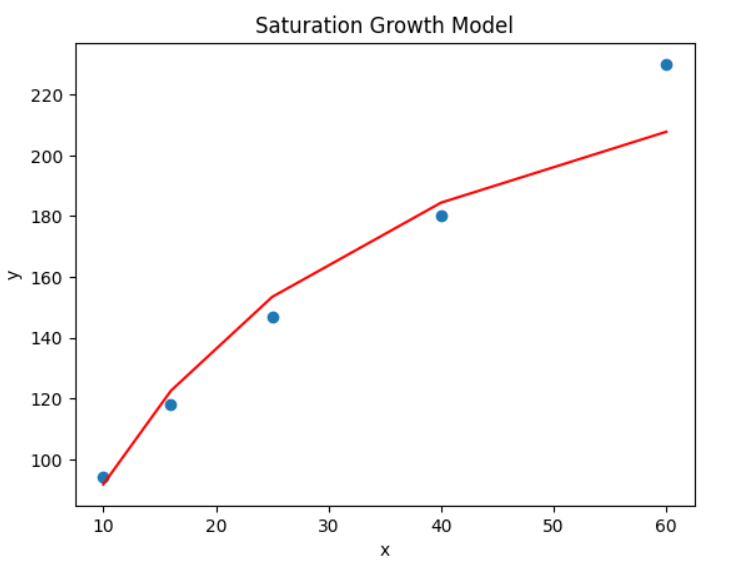
pred = []

for i in range(n):

    values = (a\*p[i])/(b+p[i])

    pred.append(values)

print(pred)



This is the graph of Power Model to predict petal length from sepal length

We get the value as 0.949

**Polynomial Regression:**

**1. Design Matrix**

The polynomial regression model extends linear regression by introducing powers of the independent variable XXX. For a cubic polynomial model, the equation takes the form:

where:

* y is the predicted petal length.
* x is the sepal length.
* β0,β1,β2,are the coefficients to be estimated.
* e is the error term.

To implement this, we first create a **Vandermonde matrix** (design matrix) for the cubic polynomial, where each column contains powers of X, from to .

def matrix(x, degree):

    X = np.vander(x, degree + 1, increasing=True)

    return X

**2. Least Squares Estimation**

We compute the polynomial regression coefficients using the **normal equation**:

where:

* X is the design matrix,
* y is the vector of observed petal lengths,
* β is the vector of regression coefficients.

This method minimizes the sum of squared residuals (the difference between observed and predicted values), giving us the best-fit polynomial curve.

def polynomial\_regression(x, y, degree):

    X = matrix(x, degree)

    coefficients = np.dot(np.linalg.inv(np.dot(X.T, X)), np.dot(X.T, y))

    return coefficients

def predict(coefficients, x):

    return np.polyval(coefficients[::-1], x)

3.Visualizing the model:

plt.scatter(x, y)

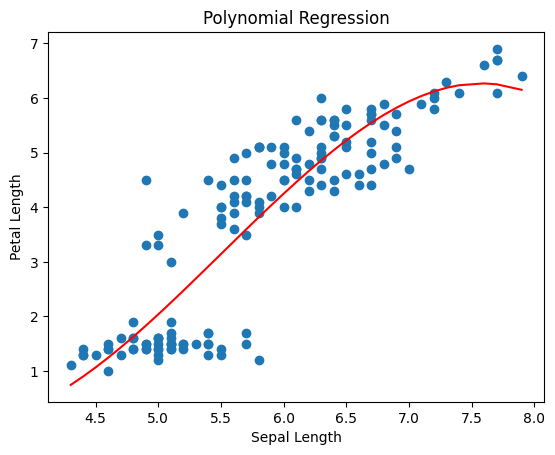
plt.plot(np.sort(x), predict(coefficients, np.sort(x)), color='red')

plt.title('Polynomial Regression')

plt.xlabel('Sepal Length')

plt.ylabel('Petal Length')

plt.show()



This is the graph of Harmonic Decline Model to predict petal length from sepal length

We get the value as 0.7829741899795442, which represents that it is a best-fit model.

**4. Evaluation Metrics**

To assess the performance of the model, we used the following metrics:

* **Mean Squared Error (MSE)**: Measures the average squared difference between actual and predicted values.
* **Mean Absolute Error (MAE)**: Measures the average absolute difference between actual and predicted values.
* **Root Mean Squared Error (RMSE)**: Provides the square root of the MSE, which is in the same units as the target variable.
* **(Coefficient of Determination)**: Measures the proportion of variance in the dependent variable explained by the independent variable.

We can implement this as

def metrics(y\_true, y\_pred):

    n = len(y\_true)

    y\_mean = np.mean(y\_true)

    ss\_total = np.sum((y\_true - y\_mean) \*\* 2)

    ss\_residual = np.sum((y\_true - y\_pred) \*\* 2)

    r\_squared = 1 - (ss\_residual / ss\_total)

    mse = np.mean((y\_true - y\_pred) \*\* 2)

    mae = np.mean(np.abs(y\_true - y\_pred))

    rmse = np.sqrt(mse)

    return mse, mae, rmse, r\_squared

**Applications of Nonlinear Regression :**

1. **Economics & Finance:**
   * Stock market forecasting and price-demand elasticity.
2. **Healthcare & Medicine:**
   * Modelling drug responses and biological growth curves.
3. **Engineering:**
   * System dynamics, material fatigue prediction, and signal processing.
4. **Environmental Science:**
   * Climate modelling and pollution impact studies.

**Conclusion:**

By carefully applying transformations, using nonlinear regression techniques, and properly choosing initial parameter estimates, you can capture the correct trends in the **exponential**, **power**, and **saturation growth** models. Each model has its specific use cases, so understanding the data and the relationship between variables is key to choosing the appropriate model and achieving accurate fits.