

Regression

Regression Models

- [Linear Regression](#)
- [Poly Regression](#)
- [Sine Regression](#)

Linear Regression

Module:

`deeprai.models.regression.linear_regression`

This module introduces a simple Linear Regression model. Linear Regression is a statistical technique commonly used for modeling and analyzing relationships between two variables.

Class: `LinearRegression`

A class representation of the linear regression model.

1. Initializer: `__init__(self)`

Description:

Initializes the `LinearRegression` class.

Attributes:

- **fitted_vals** (`list`): A list to store the results after the model has been fitted. This is primarily the coefficients of the linear equation.

Example:

```
from deeprai.models.regression.linear_regression import LinearRegression

model = LinearRegression()
```

2. Method: `fit(self, x_vals, y_vals)`

Description:

Fit the model to the given `x_vals` and `y_vals` using linear regression.

Parameters:

- **x_vals** (`list` or `np.ndarray`): The input values or features.
- **y_vals** (`list` or `np.ndarray`): The output values or labels.

Returns:

- `list`: Coefficients of the linear equation.

Example:

```
model.fit(x_vals=[1, 2, 3], y_vals=[2, 4, 6])
```

3. Method: `run(self, x_val)`

Description:

Use the previously fitted model to predict the output for a given `x_val`.

Parameters:

- **x_val** (`float`): The input value for which the prediction is desired.

Returns:

- `float`: Predicted value based on the linear regression equation.

Example:

```
predicted_val = model.run(4)
print(predicted_val)
```

Poly Regression

Module:

```
deeprai.models.regression.poly_regression
```

Class: `PolyRegression`

A class representation of the polynomial regression model.

1. **Initializer:** `__init__(self)`

Description:

Initializes the `PolyRegression` class.

Attributes:

- **fitted_vals** (`list`): A list to store the results after the model has been fitted. These values represent the coefficients of the polynomial equation.

Example:

```
from deeprai.models.regression import PolyRegression

model = PolyRegression()
```

2. **Method:** `fit(self, x_vals, y_vals)`

Description:

Fit the model to the given `x_vals` and `y_vals` using polynomial regression.

Parameters:

- **x_vals** (`list` or `np.ndarray`): The input values or features.
- **y_vals** (`list` or `np.ndarray`): The output values or labels.

Returns:

- `list`: Coefficients of the polynomial equation, starting from the coefficient of the highest degree term.

Example:

```
model.fit(x_vals=[1, 2, 3], y_vals=[2, 5, 10])
```

3. Method: `run(self, x_val)`

Description:

Use the previously fitted model to predict the output for a given `x_val` based on the polynomial equation.

Parameters:

- **x_val** (`float`): The input value for which the prediction is desired.

Returns:

- `float`: Predicted value based on the polynomial regression equation.

Example:

```
predicted_val = model.run(4)
print(predicted_val)
```

Sine Regression

Module:

```
deeprai.models.regression.sine_regression
```

Class: **SineRegression**

A class representation of the sine regression model.

1. **Initializer:** `__init__(self)`

Description:

Initializes the `SineRegression` class.

Attributes:

- **fitted_vals** (`list`): A list to store the results after the model has been fitted. These values represent the parameters of the sine equation.

Example:

```
from deeprai.models.regression import SineRegression

model = SineRegression()
```

2. **Method:** `fit(self, x_vals, y_vals)`

Description:

Fit the model to the given `x_vals` and `y_vals` using sine regression.

Parameters:

- **x_vals** (`list` or `np.ndarray`): The input values or features.
- **y_vals** (`list` or `np.ndarray`): The output values or labels.

Returns:

- `list`: Parameters of the sine equation, which includes amplitude, frequency, phase shift, and vertical shift.

Example:

```
model.fit(x_vals=[1, 2, 3], y_vals=[2, 1.5, 2.5])
```

3. Method: `run(self, x_val)`

Description:

Use the previously fitted model to predict the output for a given `x_val` based on the sine equation.

Parameters:

- **x_val** (`float`): The input value for which the prediction is desired.

Returns:

- `float`: Predicted value based on the sine regression equation.

Example:

```
predicted_val = model.run(4)
print(predicted_val)
```