ZZSC5836 Data Mining and Machine Learning

Assignment 2: Data processing and linear regression

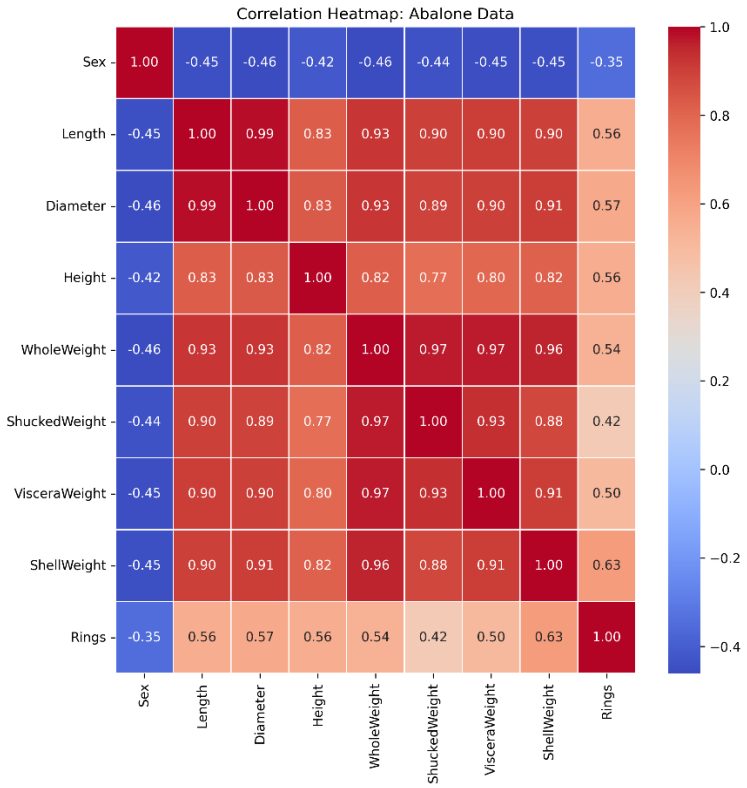
Name	Anirban Chakrabarty
ZID	z5626947
Date	23/ 03/ 2025

Table of Contents

1. Data processing

2. Major Observations from Correlation Heatmap

Below id the Correlation Heatmap generated from the Abalone Data:

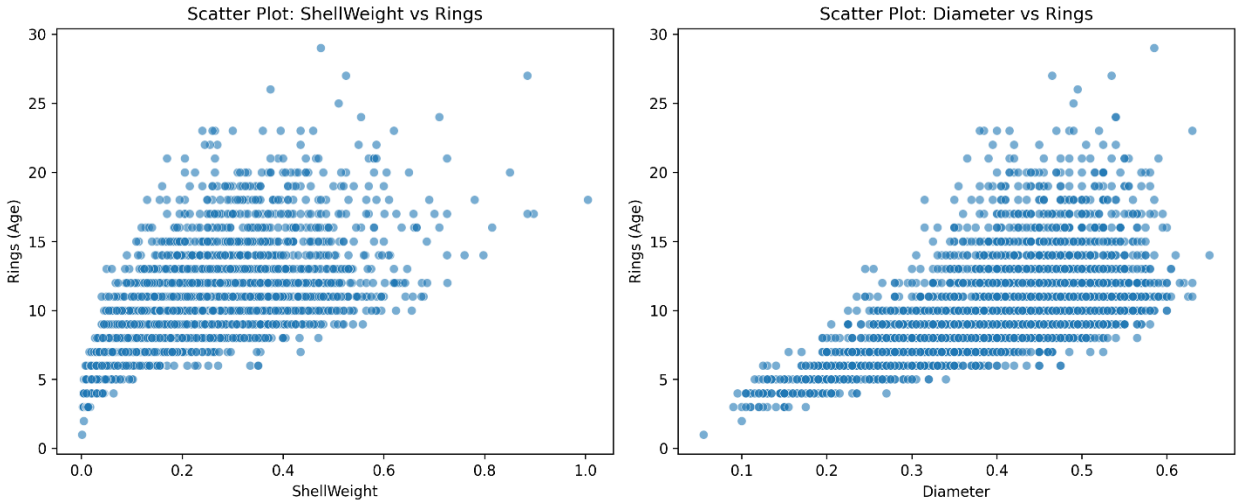


Key Observations and Interpretations are as follows

1. "Rings" (which represents **age**) has a **moderate positive correlation** with:
    - Weight parameters "WholeWeight", "VisceraWeight" and "ShellWeight".
    - Size parameters “Length” and “Diameter”.

    Therefore, **older abalones tend to be heavier and larger in size, but the relationship is not perfectly linear.**
2. Weight and Size parameters "WholeWeight", "ShuckedWeight", "VisceraWeight", "ShellWeight", “Length” and “Diameter” show high positive correlations.
    - **This implies larger abalones are likely to have larger parameters of weights, diameters and lengths**.
3. There is low or no correlation between Sex (0, 1, 2) and other continuous features like Length, Diameter, and Weight. Hence, **Sex does not significantly influence other features**.

3. Major Observations from Scatter Plots of Rings(Age) vs Two of Its Most Correlated Features



The scatter plots visualize the relationship between Rings (Age) and the two most correlated features: ShellWeight and Diameter.

ShellWeight vs. Rings (Left Plot) Observations:

1. The data points indicate a **moderately high positive correlation** between ShellWeight and Rings.
2. As ShellWeight increases, Rings (age) also tends to increase.
3. The spread of points suggests a **non-linear relationship**, with **older abalones having more variance in shell weight**.
4. **Heavier shells are associated with older abalones**.
5. However, variability increases at higher weights, meaning shell weight alone is **not a perfect predictor** of age.

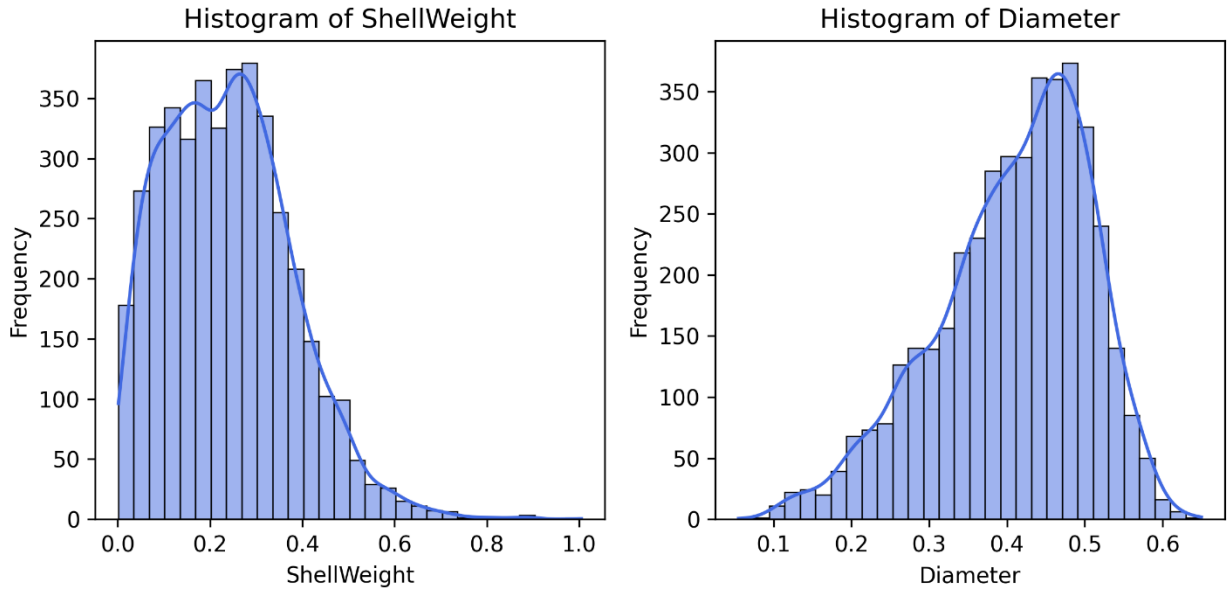
Diameter vs. Rings (Right Plot) Observations:

1. A **positive correlation** exists between Diameter and Rings, but it appears slightly weaker than ShellWeight.
2. As Diameter increases, Rings increases.
3. The spread of points suggests a more **consistent trend** compared to ShellWeight, but with **some outliers**.
4. Larger abalones tend to be older, but **diameter does not account for variations in weight or density**.
5. This feature is useful but may need to be **combined with weight-based features** for better predictions.

Key Takeaways

1. ShellWeight and Diameter are both strong predictors of age.
2. ShellWeight shows a stronger correlation, but with more variance.
3. Diameter has a more linear trend, but some deviations exist.
4. Combining these features in a model would improve age estimation.

4. Major Observations from Histograms of Two Most Correlated Features with Rings(Age)



ShellWeight Distribution:

1. The histogram is **right-skewed**, meaning most abalones have lower shell weights.
2. There are **fewer abalones with very high shell weights**.
3. The distribution suggests that heavier shells are less common, likely corresponding to older abalones (This is derived once the histogram is compared with the ShellWeight vs Rings(Age) scatter plot above)

Diameter Distribution:

1. The histogram follows a **normal-like distribution**, with a peak around **0.4 - 0.5**.
2. This suggests that most abalones have a **moderate diameter**.
3. Smaller and larger diameters are less common.

Comparison to Age (Rings):

1. Since these features are highly correlated with Rings (Age), their **right-skewed or normal distribution reflects how abalone age is distributed**.
2. Older abalones (with more rings) are likely to have **larger diameters and heavier shells**.
3. There might be some outliers in **both features**, indicating extreme values.

2. Modelling

1. Mathematical Description of the Linear Regression Model

Linear Regression is a **supervised learning algorithm** used to model the relationship between an **independent variable (features, X)** and a **dependent variable (target, Y)** by fitting a linear equation to the observed data.

1. Equation of a Simple Linear Regression Model (One Predictor)

A simple linear regression model with one predictor is given by:

$$Y = \beta_0 + \beta_1 X + \epsilon$$

Where:

- Y = **Dependent Variable (Target Output)**
- X = **Independent Variable (Feature/Input)**
- β_0 = **Intercept (Bias Term)** → Value of Y when X=0
- β_1 = **Slope (Coefficient of X)** → Change in Y per unit change in X
- ϵ = **Error Term (Residuals)** → Represents noise in data (unexplained variance)

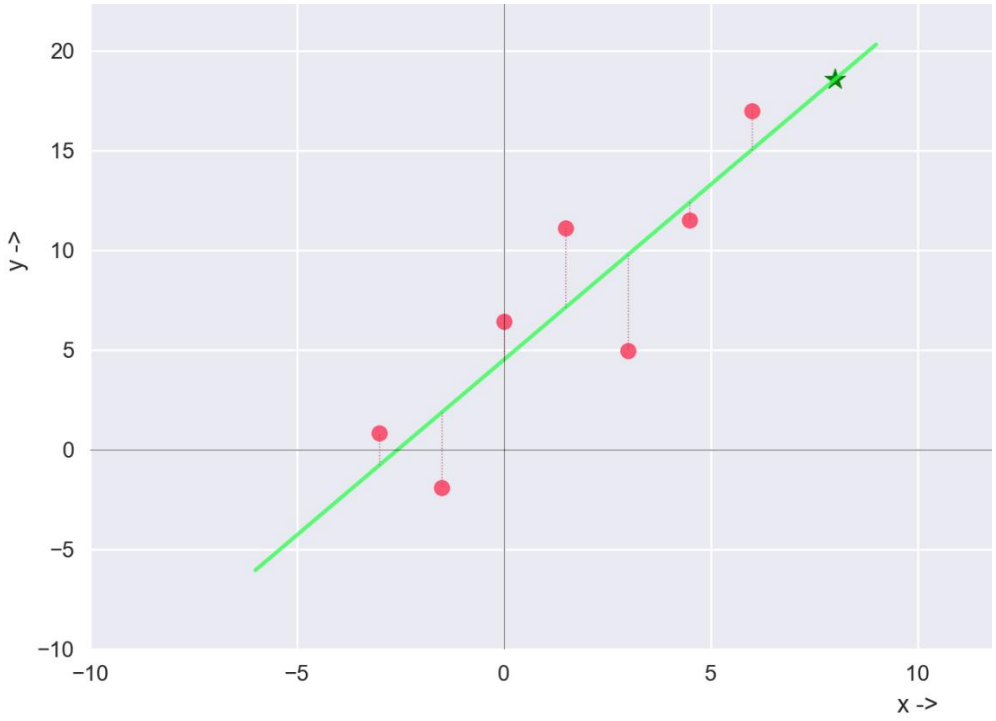


Diagram: The purpose of linear regression is to find a trend (green line above) for the scatter plot that gives the least collective error (mean squared error, indicative of the thin dotted red lines above). Further details and code are available in my GitHub repository: https://github.com/anodiamadmin/anodiam/blob/main/AnodiamContent/AI_Robotics/Youtubing/06-MathsStats/Workbooks/Statistics/07RegressionConcepts.py

2. Multiple Linear Regression (Multiple Predictors)

For multiple predictor variables $X_1, X_2, \ldots, X_n$, the model generalizes to:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n + \epsilon$$

Or, in **vector form**:

$$Y = X\beta + \epsilon$$

Where:

- Y is an m×1 vector (target values)
- X is an m×(n+1) matrix (features with a bias term)
- β is an (n+1)×1 vector (parameters to learn)
- ϵ is an m×1 error term

3. Cost Function (Mean Squared Error - MSE)

To estimate the best values of β, we minimize the error between predicted values ($\hat{Y}$) and actual values (Y). The most commonly used cost function is **Mean Squared Error (MSE)**:

$$J(\beta) = \frac{1}{m}\sum_{i=1}^{m}\left(Yi - \hat{Y}i\right)^2$$

Where *m* is the number of training data points.

We have to iteratively find out the parameters β_0 intercept, and slopes $\beta_1 \ldots \beta_n$ of $\hat{Y}$ (indicative of the trend line in light green above) so that the collective error ϵ values are minimized.

4. Gradient Descent Optimization

This is the iterative process to find out the best/ optimum values of parameters β for which the cost function above is minimized. i.e. the derivative of the cost function is = 0 (practically smaller than an acceptable value).

The values of β for which the cost function is minimized, will give the best prediction of $\hat{Y}$

Conclusion

- Linear Regression assumes a linear relationship between **features** and **target**.
- It minimizes the **Mean Squared Error (MSE)** to find optimal coefficients.
- Parameters can be estimated using **Gradient Descent**.

2. Basic Gradient Descent Algorithm

Gradient Descent is an **optimization algorithm** used to minimize the cost function and find the best parameters (β\betaβ) in **Linear Regression** and other machine learning models. It iteratively updates the parameters by taking steps proportional to the negative gradient of the cost function.

Key Points

1. **Gradient Descent iteratively minimizes the cost function** by adjusting parameters in the direction of the steepest descent.
2. **The stopping criteria**:
    - When the change in cost is very small.
    - When a predefined number of iterations is reached.
3. **The learning rate (α) is crucial**:
    - Too large → Algorithm might diverge.
    - Too small → Converges very slowly.

Algorithm Steps

1. Initialize Parameters

- Start with random values (or zeros) for the parameters $\beta 0$ intercept, and slopes $\beta_1 \ldots \beta_n$.
- Set a **learning rate** (α), which determines the step size.

2. Compute Predictions

- Calculate the predicted values:

$$\hat{Y} = X\beta$$

3. Compute Cost Function

- Use **Mean Squared Error (MSE)** to measure how well the model fits the data:

$$J(\beta) = \frac{1}{m}\sum_{i=1}^{m}\left(Yi - \hat{Y}i\right)^2$$

  where m is the number of training examples.

4. Compute Gradient (Partial Derivatives of Cost Function)

- Compute the gradient of the cost function with respect to each parameter:

$$\partial J/\partial \beta_j = \frac{-2}{m}\sum_{i=1}^{m} Xij\left(Yi - \hat{Y}i\right)$$

  This gives the direction and magnitude of the update.

5. Update Parameters

- Update each parameter using the learning rate and computed gradient:

$$\beta_j := \beta_j - \alpha\ (\partial J(\beta)/\partial \beta_j)$$

6. Repeat Until Convergence

- Repeat steps **2 to 5** until the cost function converges (i.e., changes are minimal or a maximum number of iterations is reached).

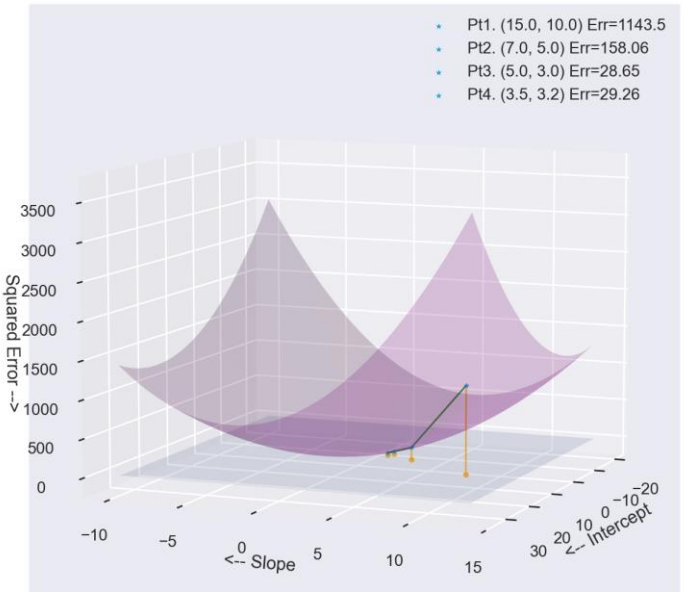


Diagram: Gradient Descent iteratively minimizing the MSE cost function for the scatter plot above.
GitHub repository:
https://github.com/anodiamadmin/anodiam/blob/main/AnodiamContent/AI_Robotics/Youtubing/06-MathsStats/Workbooks/Statistics/07RegressionConcepts.py

5. RMSE & R-squared score

Root Mean Squared Error (RMSE)

It is a measure of how well a regression model predicts an outcome. It calculates the square root of the average squared differences between actual (y_true) and predicted (y_predicted) values.

Formula:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(yi\ -\ \hat{y}i)^2}$$

Where:

- n = Total number of observations
- yi = Actual value of the i th observation
- ŷi = Predicted value of the i th observation
- $(yi\ -\ \hat{y}i)^2$ = Squared error for each observation
- ∑ => Summation over all observations
- $\frac{1}{n}\sum_{i=1}^{n}(yi\ -\ \hat{y}i)^2$ = **Mean Squared Error (MSE)**

Notes & Properties:

1. **Compute Errors** → Find the difference between actual and predicted values.
2. **Square the Errors** → Ensure all errors are positive (no cancellations).
3. **Take the Mean** → Find the average of squared errors.
4. **Take the Square Root** → Convert back to original units for better interpretability.
5. **Always non-negative**: RMSE ≥ 0.
6. **Lower RMSE = Better model fit**.
7. **Same units as the target variable** → Making interpretation easier

R-squared (R^2) Score

The R-squared score, also known as the coefficient of determination, measures how well a regression model explains the variability in the target variable. It ranges from 0 to 1, where 1 indicates a perfect fit and 0 means the model does not explain any variance.

Formula:

$$R^2 = 1 - SS_{res} / SS_{tot}$$

Where:

- Residual Sum of Squares (SS_{res}):

$$SS_{res} = \sum_{i=1}^{n} (yi - \hat{y}i)^2$$

- yi = Actual value of the i th observation
- $\hat{y}$i = Predicted value of the i th observation
- Total Sum of Squares (SS_{tot}):

$$SS_{tot} = \sum_{i=1}^{n} (yi - \bar{y}i)^2$$

- $\bar{y}$ = Mean of actual y-values

Notes & Properties:

- Measures How Much Variability is Explained by the Model:
    - SS_{tot} represents total variability in y (without using a model).
    - SS_{res} represents the variability left after using the model.
    - $R^2 = 1 - SS_{res} / SS_{tot}$ tells us how much of the total variation is explained.
- Values:
    - R2=1 → Perfect model (100% variance explained).
    - R2=0 → Model is no better than predicting the mean $\bar{y}$.
    - R2<0 → Model is worse than using the mean (indicates a very poor fit).

Comparison of Results from Different Approaches

Ideally, the results from our experiments should highlight the impact of feature normalization on the performance of the linear regression model. When using **un-normalized data**, the model should exhibit **higher variance in RMSE and R^2 scores**, indicating instability across different random splits of the dataset. The scale differences in features (e.g., weight in grams vs. length in millimeters) may affected the model's ability to converge efficiently during gradient descent. On the other hand, with **standard normalization**, where features are scaled to have a mean of zero and a standard deviation of one, the model should shows **lower variance in RMSE and R^2 scores**, leading to more consistent performance across experiments. Additionally, the **mean RMSE for normalized data should be lower**, implying better generalization, while the **mean R^2 score is higher**, suggesting a stronger explanatory power of the model. **But in case of the Abelone data we do not observe considerable lower variance in RMSE and R^2 scores with normalization. This is due to the fact that**

1. **Normalization does not affect linear regression to a great extent.**
2. **Observation of the feature data reveals that all numeric features already are of similar scale ranging from zero (0) to one (1)s. Thus Normalization does not have much effect on the features.**