import numpy as np In [1]: import pandas as pd from sklearn.model_selection import train_test_split from sklearn.preprocessing import StandardScaler from sklearn.linear_model import LinearRegression,LogisticRegress data = pd.read csv("abalone.csv") In [2]: In [3]: data Whole Shucked Viscera Shell Out[3]: Length Diameter Height Rings weight weight weight weight 0.455 0.365 0.095 0.5140 0.2245 0.1010 0.1500 15 0 M 0.350 0.265 0.090 0.2255 0.0995 0.0485 7 0.0700 2 0.530 0.135 0.6770 0.2565 0.1415 F 0.420 0.2100 9 0.2155 0.440 0.365 0.5160 3 M 0.125 0.1140 0.1550 10 0.330 0.255 0.080 0.2050 0.0895 0.0395 0.0550 7 4 П 0.565 4172 F 0.450 0.165 0.8870 0.3700 0.2390 0.2490 11 0.590 0.440 0.135 0.9660 0.4390 4173 M 0.2145 0.2605 10 4174 M 0.600 0.475 0.205 1.1760 0.5255 0.2875 0.3080 9 4175 0.625 0.485 0.150 1.0945 0.5310 0.2610 0.2960 10 0.9455 0.3765 0.4950 4176 0.710 0.555 0.195 1.9485 12 M 4177 rows × 9 columns In [4]: data.info() <class 'pandas.core.frame.DataFrame'> RangeIndex: 4177 entries, 0 to 4176 Data columns (total 9 columns): Non-Null Count Dtype # Column _____ ----- - ------0 Sex 4177 non-null object 4177 non-null 1 Length float64 2 Diameter 4177 non-null float64 4177 non-null 3 Height float64 4 Whole weight 4177 non-null float64 4177 non-null 5 Shucked weight float64 Viscera weight 4177 non-null 6 float64 Shell weight 7 4177 non-null float64 4177 non-null 8 Rings int64 dtypes: float64(7), int64(1), object(1)memory usage: 293.8+ KB pd.get_dummies(data['Sex']) In [5]: Out[5]: M 0 0 0 1 0 1 0 1 2 1 0 0 3 0 0 1 4 0 1 0 4172 1 0 0 0 4173 0 1 4174 0 0 1 4175 1 0 0 4176 0 0 1 4177 rows × 3 columns In [6]: def preprocess_and_train(df, target, task): df = df.copy()# If the sex column is not the target, one-hot encode it if target != 'Sex': dummies = pd.get_dummies(df['Sex']) df = pd.concat([df, dummies], axis=1) df = df.drop('Sex', axis=1) # Split target from df y = df[target].copy() X = df.drop(target, axis=1).copy() # Train-test split X_train, X_test, y_train, y_test = train_test_split(X, y, train) # Scale X scaler = StandardScaler() scaler.fit(X_train) X_train = pd.DataFrame(scaler.transform(X_train), columns=X.c X_test = pd.DataFrame(scaler.transform(X_test), columns=X.col # Define model if task == 'regression': model = LinearRegression() elif task == 'classification': model = LogisticRegression() model.fit(X_train, y_train) # Return the test results return model.score(X_test, y_test) In [7]: data Whole Shucked Viscera Shell Out[7]: Sex Length Diameter Height Rings weight weight weight weight 0.455 0.5140 0.2245 0.1010 0.1500 0 M 0.365 0.095 15 0.350 0.265 0.090 0.0485 7 M 0.2255 0.0995 0.0700 0.530 0.2565 2 F 0.420 0.135 0.6770 0.1415 0.2100 9 M 0.440 0.365 0.125 0.5160 0.2155 0.1140 0.1550 10 0.330 0.255 0.080 0.2050 0.0895 0.0395 0.0550 7 4 П ... 0.3700 4172 0.565 0.450 0.165 0.8870 0.2390 0.2490 F 11 0.590 0.440 4173 M 0.135 0.9660 0.4390 0.2145 0.2605 10 0.205 0.5255 4174 M 0.600 0.475 1.1760 0.2875 0.3080 9 4175 0.625 0.485 0.150 1.0945 0.5310 0.2610 0.2960 10 4176 0.710 0.555 0.195 1.9485 0.9455 0.3765 0.4950 12 M 4177 rows × 9 columns results = preprocess_and_train(data, target='Sex', task='classifi In [8]: print("Sex Classification Accuracy: {:.2f}%".format(results * 100 Sex Classification Accuracy: 57.10% data [9]: Whole Shucked Viscera Shell Out[9]: Sex Length Diameter Height Rings weight weight weight weight 0 M 0.455 0.365 0.095 0.5140 0.2245 0.1010 0.1500 15 1 M 0.350 0.265 0.090 0.2255 0.0995 0.0485 0.0700 7 0.1415 2 0.530 0.420 0.135 0.6770 0.2565 0.2100 F 9 0.440 0.125 3 M 0.365 0.5160 0.2155 0.1140 0.1550 10 0.255 0.0895 0.0395 7 4 ١ 0.330 0.080 0.2050 0.0550 4172 F 0.565 0.450 0.165 0.8870 0.3700 0.2390 0.2490 11 0.590 4173 M 0.440 0.135 0.9660 0.4390 0.2145 0.2605 10 0.600 0.475 0.28754174 M 0.205 1.1760 0.5255 0.3080 9 0.485 0.150 4175 0.625 1.0945 0.5310 0.2610 0.2960 10 4176 M 0.710 0.555 0.195 1.9485 0.9455 0.3765 0.4950 12 4177 rows × 9 columns results = preprocess_and_train(data, target='Length', task='regre In [10]: print("Length Regression R^2: {:.4f}".format(results)) Length Regression R^2: 0.9753 data [11]: Whole Shucked Viscera Shell Out[11]: Diameter Sex Length Height Rings weight weight weight weight 0.365 0.095 0 M 0.455 0.5140 0.2245 0.1010 0.1500 15 7 1 M 0.350 0.265 0.090 0.2255 0.0995 0.0485 0.0700 2 0.530 0.420 0.135 0.6770 0.2565 0.1415 F 0.2100 9 3 M 0.440 0.365 0.125 0.5160 0.2155 0.1140 0.1550 10 0.0395 7 4 ١ 0.330 0.255 0.080 0.2050 0.0895 0.0550 4172 F 0.565 0.450 0.165 0.8870 0.3700 0.2390 0.2490 11 4173 M 0.590 0.440 0.135 0.9660 0.4390 0.2145 0.2605 10 4174 M 0.600 0.475 0.205 1.1760 0.5255 0.2875 0.3080 9 4175 0.625 0.485 0.150 1.0945 0.5310 0.2610 0.2960 10 0.555 0.195 1.9485 0.9455 0.3765 0.4950 12 4176 M 0.710 4177 rows × 9 columns results = preprocess_and_train(data, target='Diameter', task='reg Ιn [12]: print("Diameter Regression R^2: {:.4f}".format(results)) [13]: Ιn Diameter Regression R^2: 0.9758 data In [14]: Whole **Shucked** Viscera Shell Out[14]: Sex Length Diameter Height Rings weight weight weight weight 0 M 0.455 0.365 0.095 0.5140 0.2245 0.1010 0.1500 15 7 1 M 0.350 0.265 0.090 0.2255 0.0995 0.0485 0.0700 0.440 0.365 0.125 0.5160 0.2155 3 M 0.11400.155010 0.330 4 ١ 0.255 0.080 0.2050 0.0895 0.0395 0.0550 7 4172 F 0.565 0.450 0.165 0.8870 0.3700 0.2390 0.2490 11 0.590 0.135 0.4390 0.2145 0.2605 4173 M 0.4400.9660 10 0.600 0.205 0.5255 0.2875 4174 M 0.475 1.1760 0.3080 9 F 0.625 0.485 0.150 0.5310 0.2610 0.2960 4175 1.0945 10 0.710 0.555 0.195 1.9485 0.9455 0.3765 0.4950 4176 M 12 4177 rows × 9 columns results = preprocess_and_train(data, target='Height', task='regre In [15]: print("Height Regression R^2: {:.4f}".format(results)) Height Regression R^2: 0.8147 In [16]: data Whole **Shucked** Viscera Shell Out[16]: Sex Length Diameter Height Rings weight weight weight weight 0.5140 0 M 0.455 0.365 0.095 0.2245 0.1010 0.1500 15 0.350 0.265 0.090 0.2255 0.0485 0.0700 7 1 M 0.0995 0.2565 0.530 0.420 0.135 0.6770 0.1415 9 2 F 0.2100 3 M 0.440 0.365 0.125 0.5160 0.2155 0.1140 0.1550 10 4 ١ 0.330 0.255 0.080 0.2050 0.0895 0.0395 0.0550 7 4172 0.565 0.2390 F 0.450 0.165 0.8870 0.3700 0.2490 11 4173 0.590 0.135 0.4390 0.2145 0.2605 M 0.440 0.9660 10 0.5255 0.600 0.475 0.205 1.1760 0.2875 4174 M 0.3080 9 F 0.625 0.485 0.150 0.5310 0.2610 0.2960 4175 1.0945 10 4176 0.710 0.555 0.195 1.9485 0.9455 0.3765 0.4950 M 12 4177 rows × 9 columns In [17]: results = preprocess_and_train(data, target='Whole weight', task= print("Whole weight Regression R^2: {:.4f}".format(results)) Whole weight Regression R^2: 0.9908 In [18]: data Whole Out[18]: **Shucked** Viscera Shell Sex Length Diameter Height Rings weight weight weight weight 0.455 0.365 0 M 0.095 0.5140 0.2245 0.1010 0.1500 15 0.0700 1 M 0.350 0.265 0.090 0.2255 0.0995 0.0485 7 2 F 0.530 0.420 0.135 0.6770 0.2565 0.1415 0.2100 9 0.440 0.125 0.2155 3 M 0.365 0.5160 0.1140 0.1550 10 0.080 4 I 0.330 0.255 0.2050 0.0895 0.0395 0.0550 7 0.565 0.165 0.2490 4172 F 0.450 0.8870 0.3700 0.2390 11 0.4404173 M 0.590 0.135 0.9660 0.4390 0.2145 0.2605 10 0.600 0.475 0.205 0.5255 0.2875 4174 1.1760 0.3080 0.2610 4175 0.625 0.485 0.150 1.0945 0.5310 0.2960 10 0.555 0.195 1.9485 0.3765 0.4950 4176 M 0.710 0.9455 12 4177 rows × 9 columns results = preprocess_and_train(data, target='Shucked weight', tas In [19]: print("Shucked-Weight Regression R^2: {:.4f}".format(results)) Shucked-Weight Regression R^2: 0.9676 data In [20]: Out[20]: Whole Shucked Viscera Shell Rings Sex Length Diameter Height weight weight weight weight 0 M 0.455 0.365 0.095 0.5140 0.2245 0.1010 0.1500 15 0.350 0.090 7 1 M 0.265 0.2255 0.0995 0.0485 0.0700 0.420 0.135 0.6770 0.2565 2 F 0.530 0.1415 0.2100 9 0.365 0.125 0.2155 3 M 0.440 0.5160 0.1140 0.1550 10 0.0395 4 I 0.330 0.255 0.080 0.2050 0.0895 0.0550 7 0.565 0.165 0.2390 4172 F 0.450 0.8870 0.3700 0.2490 11 0.590 0.2605 4173 M 0.440 0.135 0.9660 0.4390 0.2145 10 0.600 0.205 0.3080 4174 M 0.475 1.1760 0.5255 0.2875 9 F 4175 0.625 0.485 0.150 1.0945 0.5310 0.2610 0.2960 10 0.3765 4176 M 0.710 0.555 0.195 1.9485 0.9455 0.4950 12 4177 rows × 9 columns In [21]: results = preprocess_and_train(data, target='Viscera weight', tas print("Viscera-Weight Regression R^2: {:.4f}".format(results)) Viscera-Weight Regression R^2: 0.9462 data In [22]: Shell Out[22]: Whole Shucked Viscera Height Diameter Rings Length Sex weight weight weight weight 0 0.455 0.365 0.095 0.5140 0.2245 0.1010 0.1500 15 M 0.350 7 1 0.265 0.090 0.2255 0.0995 0.0485 0.0700 M 0.6770 2 0.530 0.420 0.135 0.2565 0.1415 0.2100 9 F M 0.440 0.365 0.125 0.5160 0.2155 0.1140 0.1550 10 3 4 0.330 0.255 0.080 0.2050 0.0895 0.0395 0.0550 7 0.2490 4172 F 0.565 0.450 0.165 0.8870 0.3700 0.2390 11 0.590 4173 M 0.440 0.135 0.9660 0.4390 0.2145 0.2605 10 4174 0.600 0.475 0.205 1.1760 0.5255 0.2875 0.3080 9 0.5310 0.2960 4175 F 0.625 0.485 0.150 1.0945 0.2610 10 1.9485 4176 0.710 0.555 0.195 0.9455 0.3765 0.4950 12 M 4177 rows × 9 columns results = preprocess_and_train(data, target='Shell weight', task= In [23]: print("Shell-Weight Regression R^2: {:.4f}".format(results)) Shell-Weight Regression R^2: In [24]: data Whole Out[24]: Shucked Viscera Shell Length Height **Rings** Sex Diameter weight weight weight weight 0.455 0.365 0.095 0.5140 0.2245 0.1010 0.1500 0 M 15 1 0.350 0.265 0.090 0.2255 0.0995 0.0485 0.0700 7 M 2 F 0.530 0.420 0.135 0.6770 0.2565 0.1415 0.2100 9 0.2155 0.125 0.1140 M 0.440 0.365 0.5160 0.1550 3 10 4 0.330 0.255 0.0895 0.0395 7 ١ 0.080 0.2050 0.0550 0.165 4172 F 0.565 0.450 0.8870 0.3700 0.2390 0.2490 11 0.2145 0.590 0.4400.135 0.9660 0.43900.2605 4173 M 10 4174 0.600 0.475 0.205 1.1760 0.5255 0.2875 0.3080 9 M F 0.625 0.485 0.150 0.5310 0.2610 0.2960 4175 1.0945 10 0.195 0.710 0.555 1.9485 0.9455 0.3765 0.4950 4176 M 12 4177 rows × 9 columns In [25]: results = preprocess_and_train(data, target='Rings', task='regres print("Rings Regression R^2: {:.4f}".format(results)) Rings Regression R^2: 0.5196 In [26]: data Whole **Shucked** Viscera Shell Out[26]: Sex Length Diameter Height Rings weight weight weight weight 0.455 0.365 0.095 0.5140 0.2245 0.1500 0 M 0.1010 15 1 0.350 0.265 0.090 0.2255 0.0995 0.0485 0.0700 7 M F 0.530 0.420 0.135 0.6770 0.2565 0.1415 0.2100 9 2 0.125 0.5160 0.2155 3 M 0.440 0.365 0.1140 0.1550 10 4 ١ 0.330 0.255 0.080 0.2050 0.0895 0.0395 0.0550 7 0.165 0.2390 4172 0.565 0.450 0.3700 F 0.8870 0.2490 11 0.2145 0.590 0.4400.135 0.9660 0.43900.2605 4173 M 10 4174 0.600 0.475 0.205 0.5255 0.2875 0.3080 M 1.1760 9 F 0.625 0.485 0.150 0.5310 0.2610 0.2960 4175 1.0945 10 0.710 0.555 0.195 1.9485 0.9455 0.3765 0.4950 4176 M 12 4177 rows × 9 columns In [27]: results = preprocess_and_train(data, target='Rings', task='classi print("Rings Classification Accuracy: {:.2f}%".format(results * Rings Classification Accuracy: 25.92% C:\Users\kisho\anaconda3\lib\site-packages\sklearn\linear_model _logistic.py:814: ConvergenceWarning: lbfgs failed to converge (status=1): STOP: TOTAL NO. of ITERATIONS REACHED LIMIT. Increase the number of iterations (max_iter) or scale the data a https://scikit-learn.org/stable/modules/preprocessing.html Please also refer to the documentation for alternative solver op tions: https://scikit-learn.org/stable/modules/linear_model.html#lo gistic-regression n_iter_i = _check_optimize_result(