Data Mining and Adv. Statistical Modeling Mini-Project Report

Jeeva Mary Loui

Metro College Of Technology

Linear Regression Using Python

Predict The age of Abalone

Abstract

Abalone are a type of shellfish that are common along the coasts of most continents across the world. By cutting through the shell of an abalone, its age can be determined by counting the number of rings using a microscope, very similar to the process used for tree rings. However, the age may also be predicted by considering a number of explanatory factors, which is a much less time-consuming process.Data collected from the physical measurements of Abalone to develop a linear regression model to determine the age of abalone through this explanatory factors

Predict The age of Abalone using Linear Regression

Dataset

Abalone and its importance

Abalone is common name for any group of small to very large sea snails, commonly found along the coasts across the world, and used as delicacy in cuisines and it's leftover shell is fashioned into jewelry due to its iridescent luster. Due to its demand and economic value it's often harvested in farms, and as such the need to predict the age of abalone from physical measurements. Traditional approach to determine its age is by cutting the shell through the cone, staining it, and counting the number of rings through a microscope -- a boring and time-consuming task.

Data Description

- Number of instances: 4177
- Number of attributes: 8
- Features: Sex, Length, Diameter, Height, Whole weight, Shucked weight, Viscera weight, and Shell weight
- Target: Rings

Note: Number of rings is the value to predict

Dataset source

Dataset comes from UCI Machine Learning repository:

https://archive.ics.uci.edu/ml/datasets/Abalone

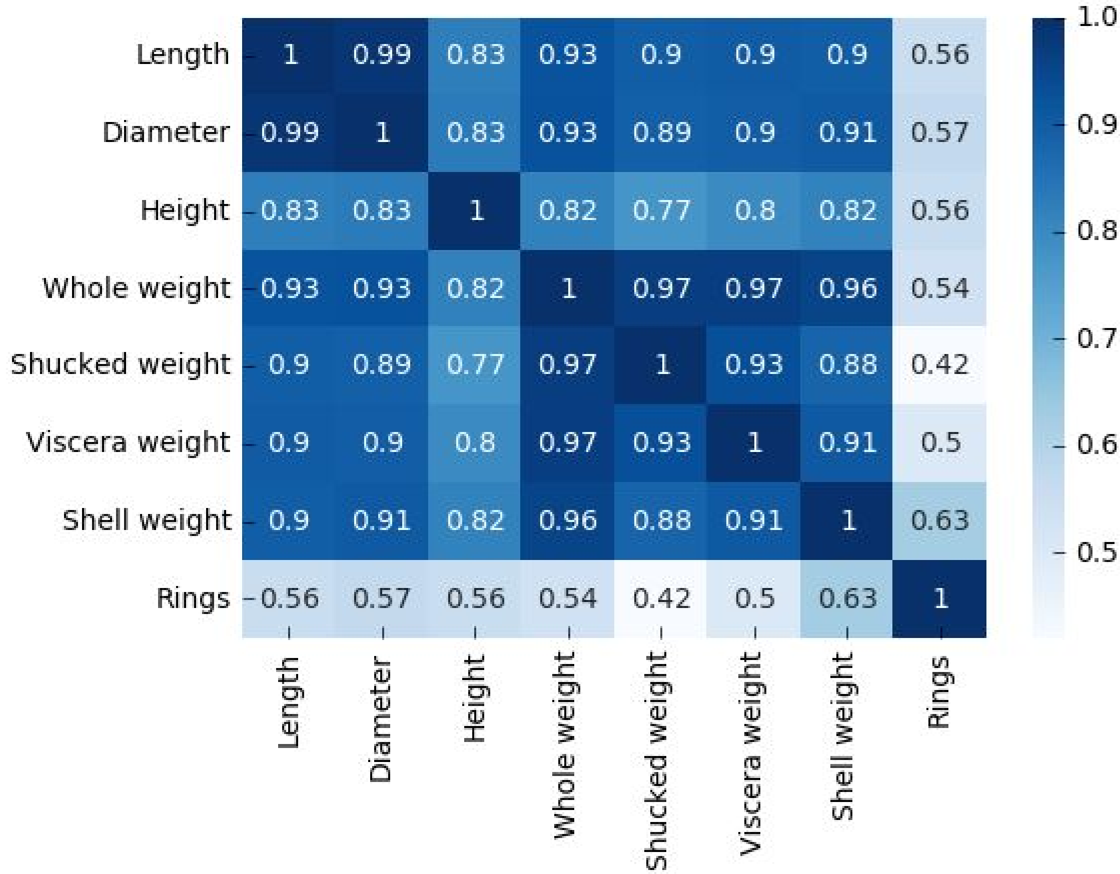
Data Preprocessing

In order to do Linear regression to predict the age there has to be done some preprocessing.

Index	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	M	0.455	0.365	0.095	0.514	0.2245	0.101	0.15	15
1	M	0.35	0.265	0.09	0.2255	0.0995	0.0485	0.07	7
2	F	0.53	0.42	0.135	0.677	0.2565	0.1415	0.21	9
3	M	0.44	0.365	0.125	0.516	0.2155	0.114	0.155	10
4	I	0.33	0.255	0.08	0.205	0.0895	0.0395	0.055	7

```
dataset=pd.read_csv('abalone.csv')
description=dataset.describe()
dataset.dtypes
#Handling missig values
dataset.isnull().sum()
#Correlation analysis of numerical variables
dataset.corr()
sns.heatmap(dataset.corr(),cmap='Blues' , annot= True)
#Encoding
dataset['Sex'] = dataset['Sex'].map( {'M':1, 'F':2 , 'I':0} )
#Extract independant and response variables
X= dataset.drop(['Rings'], axis=1)
Xs=X
y = dataset['Rings'].reshape(-1,1)
```

There are no missing values and the only non numerical attribute is Sex . It is encoded to numerical using map function in pandas.A correlation analysis is done to understand the dependant and response attributes.



It is clear from this heatmap of correlation between the numerical attributes of the dataset that ‘Rings’ is the response attribute,so it is extracted as independent and response variable from the dataset.

The next step is to do normalisation to do the linear regression. Standardscaling is imported from scikit learn library to do the normalisation or scaling.

```
#Normalise
from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
sc_y = StandardScaler()
X = sc_X.fit_transform(X)
y = sc_y.fit_transform(y)
```

Now it's important to check if any attribute can be eliminated to do the regression so the model is much efficient, the technique I chose to use is Recursive Feature Elimination

```
#Recursive Feature Elimination
from sklearn.linear_model import LinearRegression
from sklearn.feature_selection import RFE
adj_R2 = []
feature_set = []
max_adj_R2_so_far = 0
n = len(X)
k = len(X[0])
for i in range(1,k+1):
    selector = RFE(LinearRegression(), i,verbose=1)
    selector = selector.fit(X, y)
    current_R2 = selector.score(X,y)
    current_adj_R2 = 1-(n-1)*(1-current_R2)/(n-i-1)
    adj_R2.append(current_adj_R2)
    feature_set.append(selector.support_)
    if max_adj_R2_so_far < current_adj_R2:
        max_adj_R2_so_far = current_adj_R2
        selected_features = selector.support_
    print('End of iteration no. {}'.format(i))
print(selected_features)
X_sub = X[:,selected_features]
```

The selected feature is boolean list : [True False True True True True True True] , Here the second attribute which is the Diameter of the Abalone is found as a feature of least importance to predict the age through Recursive Feature Elimination

Train and Build a Linear Regression Model

The dataset has to be split into the train and test to train a regression model and test the model using the test set. The cross_validation in scikit package helps doing this efficiently. The training set is fit to a linear regression model. The model coefficients are determined as well.

```
#Spliting of dataset
from sklearn.cross_validation  import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_sub,y,random_state=0)
#train the model
model = LinearRegression()
model.fit(X_train,y_train)
model.coef_
```

In [41]:

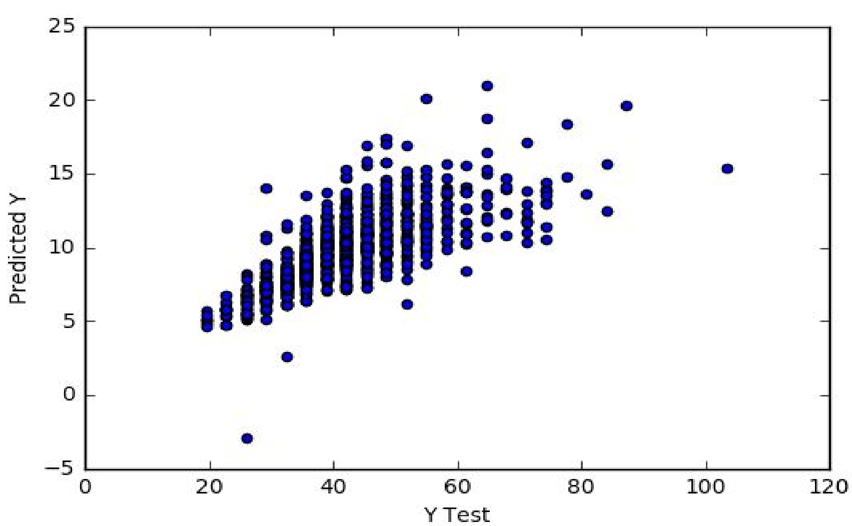
model.coef_ : array([[0.0942248 , 0.34698737, 0.10333993, 1.42161099, -1.39783053, -0.3628958 , 0.38364238]]) , which represents the mean change in the response variable for one unit of change in the predictor variable.

Performance Analysis

Now the model is ready and model performance has to evaluated .The model score is only 0.539 which is not that exciting.

```
#see performance score
model.score(X_test,y_test)
#prediction
y_pred = model.predict(X_test)
y_pred = sc_y.inverse_transform(y_pred.reshape(len(y_pred),1)).reshape(len(y_pred))
y_test = sc_y.inverse_transform(y_test.reshape(len(y_test),1)).reshape(len(y_test))
plt.scatter(y_test,y_pred)
plt.xlabel('Y Test')
plt.ylabel('Predicted Y')
#see performance score
from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
import statsmodels.api as sm
#OLS model
X_modified = sm.add_constant(X_train)
lin_reg = sm.OLS(y_train,X_modified)
result = lin_reg.fit()
print(result.summary())
```

The test data is normalised to do the modelling and for performance analysis it is rescaled to that of original data using inverse_transform. A scatter plot is used to see the trained model and is tested using the test data set aside



.

The predicted model has some degree of scattering which has been strengthen by the weak model score.

Performance Scores:

Root Mean Squared Error: 2.20067833083

Our model was able to predict the number of rings of every abalone in the test set within 2.20067833083 of the real number.

Mean Absolute Error: 1.5847690276

Mean Squared Error: 4.84298511579

```
OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.531
Model:                            OLS   Adj. R-squared:                  0.529
Method:                 Least Squares   F-statistic:                     504.3
Date:                Sun, 26 May 2019   Prob (F-statistic):               0.00
Time:                        19:57:55   Log-Likelihood:                -3253.7
No. Observations:                3132   AIC:                             6523.
Df Residuals:                    3124   BIC:                             6572.
Df Model:                           7
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         -0.0024      0.012     -0.193      0.847      -0.026       0.022
x1             0.0942      0.015      6.484      0.000       0.066       0.123
x2             0.3470      0.035      9.847      0.000       0.278       0.416
x3             0.1033      0.021      4.889      0.000       0.062       0.145
x4             1.4216      0.124     11.439      0.000       1.178       1.665
x5            -1.3978      0.064    -21.841      0.000      -1.523      -1.272
x6            -0.3629      0.051     -7.166      0.000      -0.462      -0.264
x7             0.3836      0.055      7.028      0.000       0.277       0.491
==============================================================================
Omnibus:                      681.069   Durbin-Watson:                   1.960
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             1731.876
Skew:                           1.174   Prob(JB):                         0.00
Kurtosis:                       5.786   Cond. No.                         28.4
```

From OLS regression results R-Squared value is 0.531 so in our model 53.1% of the variability in Y can be explained using X. This is not that exciting.The adjusted R-squared compares the explanatory power of regression models that contain different numbers of predictors and its is also not a promising score of 52% only.

Conclusion

The Abalone Dataset has been pre processed efficiently and Linear Model is fit with the data and the age abalone can be predicted using this model. The performance scores tells that model is not that exciting and errors might occur with the available predictor variables of the response which is the number of rings.