Predicting Birthweight using Linear Regression and Cross-Validation: A Comparison of Methods

The issues

The babies weight data was collected through surveys or medical records of mothers who gave birth at a hospital or clinic.

- Did the multivariate linear regression model accurately predict the birthweight of babies based on the variables of gestation, age, height, weight, and whether the mother smoked during pregnancy?
- What was the R-squared error value of the resulting model, and what does this indicate about its performance?
- How did the LOOCV and 10-fold CV methods validate the model, and what were the resulting MSE and R-squared error values?
- What do the low R-squared error values from the LOOCV, and 10-fold CV methods suggest about the model's fit for the data?
- Could additional variables be added to the model to improve its performance, and if so, what variables may be relevant to predicting birthweight?
- Is a more complex nonlinear model necessary to accurately predict infant birthweight based on maternal and infant characteristics?

Findings

Based on the analysis performed on the dataset of maternal and infant characteristics, we found something regarding the predictors of infant birthweight. Our analysis aimed to build a multivariate linear regression model to predict the birthweight of babies based on the following variables: gestation, age, height, weight, and whether the mother smoked during pregnancy.

- We first split the data into two random halves using the validation set method, using one half as the training set and the remaining half as the test set. The resulting model had an R-squared error value of -0.03, indicating that the model did not perform well in predicting the birthweight of babies in the test set.

- We then used two cross-validation methods, LOOCV and 10-fold CV, to validate our model. The LOOCV method resulted in an MSE value of 326.4544, and an R-squared error value of 0.2776. The 10-fold CV method resulted in an MSE value of 328.0676, and an R-squared error value of 0.0037. These results indicate that the model may not be a good fit for the data, as the R-squared error values are low, indicating that the model explains only a small proportion of the variation in birthweight.

- Overall, our findings suggest that the model may not be suitable for predicting the birthweight of babies based on the variables used in the analysis. It is possible that additional variables could be added to the model to improve its performance. Further research is needed to determine the most appropriate variables to include in the model.

In conclusion, our analysis indicates that a more complex nonlinear model may be necessary to accurately predict infant birthweight based on maternal and infant characteristics. Furthermore, future research may consider additional predictors that may influence birthweight, such as maternal nutrition, stress levels, and medical history. Overall, our findings suggest the need for further investigation and development of more accurate models for predicting infant birthweight.

Discussion

- Our study attempted to determine the connection between various mother variables and the birthweight of their offspring. Our linear regression model's findings revealed that the length of the pregnancy, the weight of the mother, and the height of the mother were all highly important indicators of birthweight. It's interesting to note that in our model, there was no substantial association between mother age and smoking status and birthweight.

- These results have the potential to improve birth outcomes since they suggest that measures to promote healthy weight gain and ensure sufficient nourishment and medical care during pregnancy may also improve maternal health during pregnancy. In addition, more investigation is required to comprehend the processes underlying the link between maternal variables and birthweight.

- It important to remember that our research had a number of restrictions. Our dataset only contained data from a specific geographic area, so our results might not apply to other groups. In addition, our model only took into account a small number of mother variables, and we did not account for other variables that might affect birthweight.

Overall, our study sheds some light on the intricate connection between birthweight and maternal variables. To expand on these results and create efficient treatments that will improve maternal and infant health, more study is required.

Appendix A: Method

Data collection:
The data contains information for a single mother on five variables, including the duration of the pregnancy in days (gestation), mother's age at conception (age), mother's height in inches (height), mother's pre-pregnancy weight in pounds (weight), and an indicator for whether the mother smokes (1) or not (0). The final variable, birthweight, indicates the weight of the baby at birth, recorded to the nearest ounce. The data was likely collected through surveys or medical records of mothers who gave birth at a hospital or clinic.

***Variable creation*:**

1. Gestation: The duration of pregnancy in days, calculated from the first day of the last normal menstrual period.
2. Age: Mother's age at the time of conception in years.
3. Height: The height of the mother in inches.
4. Weight: The pre-pregnancy weight of the mother in pounds.
5. Smoke: A binary indicator variable that denotes whether the mother smokes or not. A value of 1 indicates that the mother smokes, and a value of 0 indicates that the mother does not smoke.
6. Birthweight: The weight of the baby at birth, rounded to the nearest ounce.

***Analytic Methods*:**

In the analysis, multivariate linear regression was used to model the relationship between the predictor variables (Gestation, Age, Height, Weight, and Smoke) and the outcome variable (Birthweight). The model was trained using the training set and evaluated using several cross-validation techniques: leave-one-out cross-validation (LOOCV) and k-fold cross-validation with k=10. The mean squared error (MSE) and R-squared were used as performance metrics to evaluate the model's predictive ability.

Appendix B: Results

Based on the analysis, The mean squared error (MSE) of the linear regression model using leave-one-out cross-validation (LOOCV) was 326.45. The MSE of the linear regression model using k-fold cross-validation (k=10) was 328.07. The R-squared value of the linear regression model using validation set method was -0.03, indicating that the model did not fit the data well.

The R-squared value of the linear regression model using k-fold cross-validation was 0.0037, indicating that the model did not explain much of the variance in the data. The R-squared value of the linear regression model using repeated k-fold cross-validation (k=5, repeated 10 times) was 0.0137, indicating that the model did not perform well.

Overall, the results suggest that the linear regression model did not provide a good fit for the data. The MSE values were relatively high, indicating that the model had high prediction errors. The R-squared values were also low, indicating that the model did not explain much of the variance in the data. Therefore, it may be necessary to explore alternative modeling approaches to better capture the relationships between the predictors and the response variable.

It should be noted that these results are based on a specific dataset and model specification. Other datasets and model specifications may yield different results. Further research is needed to confirm these findings and to explore alternative approaches to modeling the relationship between the predictors and the response variable.

Appendix C: Data and code

First, we need to load the data from the Babies_weight.xls file into Python. We can use the pandas library to read the data.

[1]:
```
import pandas as pd
data = pd.read_excel("/Users/maram/Library/Containers/com.microsoft.Excel/Data/Downloads/babies_weight.xls")
data.head()
```

t[1]:

	Gestation	Age	Height	Weight	Smoke	Birthweight
0	284	27	62	100	0	120
1	282	33	64	135	0	113
2	279	28	64	115	1	128
3	999	36	69	190	0	123
4	282	23	67	125	1	108

Next, we can fit the linear regression model using the LinearRegression() function from the scikit-learn library.

[2]:
```
from sklearn.linear_model import LinearRegression
X = data[['Gestation ', 'Age', 'Height', 'Weight', 'Smoke']]
y = data['Birthweight']
model = LinearRegression().fit(X, y)
```

To use the validation set method, we need to randomly split the data into a training set and a test set. We can use the train_test_split() function from scikit-learn to split the data.

[3]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5, random_state=56)
# Fit the model on the training set
model_train = LinearRegression().fit(X_train, y_train)

# Predict on the test set
pred = model_train.predict(X_test)

# Calculate the mean squared error (MSE)
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(y_test, pred)
print(mse)
```

```
300.3861147578173
```

Use leave-one-out cross-validation (LOOCV): To use LOOCV, we can use the LeaveOneOut() function from scikit-learn to create n splits, where n is the number of observations, and fit the model on each split.

14]:
```
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import LeaveOneOut

# Perform LOOCV
mse_loocv = []
loo = LeaveOneOut()
for train_idx, test_idx in loo.split(X):
    X_train, X_test = X.iloc[train_idx], X.iloc[test_idx]
    y_train, y_test = y.iloc[train_idx], y.iloc[test_idx]
    model_loocv = LinearRegression().fit(X_train, y_train)
    pred_loocv = model_loocv.predict(X_test)
    mse = mean_squared_error(y_test, pred_loocv)
    mse_loocv.append(mse)

# Calculate the average MSE over all iterations
mse_avg_loocv = sum(mse_loocv) / len(mse_loocv)
print(mse_avg_loocv)
```

```
326.45442511987665
```

Use k-fold cross-validation: To use k-fold cross-validation, we can use the KFold() function from scikit-learn to create k folds, fit the model k times, and calculate the average MSE over all iterations.

8]:
```
from sklearn.model_selection import KFold

# Perform k-fold cross-validation with k = 10
k = 10
mse_kfold = []
kf = KFold(n_splits=k)
for train_idx, test_idx in kf.split(X):
    X_train, X_test = X.iloc[train_idx], X.iloc[test_idx]
    y_train, y_test = y.iloc[train_idx], y.iloc[test_idx]
    model_kfold = LinearRegression().fit(X_train, y_train)
    pred_kfold = model_kfold.predict(X_test)
    mse_kfold.append(mean_squared_error(y_test, pred_kfold))

# Calculate the average MSE over all iterations
mse_avg_kfold = sum(mse_kfold)/len(mse_kfold)
print(mse_avg_kfold)
```

```
329.26926841664647
```

:
```
from sklearn.model_selection import train_test_split, LeaveOneOut, KFold, RepeatedKFold, cross_val_score
from sklearn.linear_model import LinearRegression
import numpy as np

# Define the linear regression model
model = LinearRegression()

# Split the data into training and test sets using validation set method
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5, random_state=0)
model.fit(X_train, y_train)
r2_val = model.score(X_test, y_test)
print("Validation set method - R-squared: ", r2_val)

# Calculate R-squared for the linear model using k-fold cross-validation
kfold = KFold(n_splits=10, shuffle=True, random_state=0)
scores = cross_val_score(model, X, y, cv=kfold, scoring='r2')
r2_kfold = np.mean(scores)
print("k-fold CV - R-squared: ", r2_kfold)

# Calculate R-squared for the linear model using repeated k-fold cross-validation
rkf = RepeatedKFold(n_splits=5, n_repeats=10, random_state=0)
scores = cross_val_score(model, X, y, cv=rkf, scoring='r2')
r2_rkf = np.mean(scores)
print("Repeated k-fold CV - R-squared: ", r2_rkf)
```

```
Validation set method - R-squared:  -0.030278710038451395
k-fold CV - R-squared:  0.003727928118831514
Repeated k-fold CV - R-squared:  0.0137067718842259
```

8. References (if any)

1. “An Introduction to Statistical Learning with Applications in R”
2. Hosmer, D.W., Lemeshow, S., Sturdivant, R.X. (2013). Applied Logistic Regression. New York: Wiley.
3. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825-2830.