

Predicting Daily Active Energy Burned: Ridge Regression Analysis

by Micah Borghese

The modern lifestyle often leads to sedentary habits, which can negatively impact one's health. Therefore, understanding and predicting daily energy expenditure becomes crucial for individual health monitoring and management. This project uses personal health data collected from an Apple Watch to predict daily active energy burned (Cal). The objective is to find a reliable way to estimate energy expenditure from daily activities, which can then inform personal health and fitness decisions.

Data Sourcing

The dataset for this analysis comes from my personal Apple Watch, which records various health metrics throughout the day. Each observation corresponds to one day of data, including the features:

- Distance walking/running (mi)
- Exercise time (min)
- Flights climbed (count)
- Heart rate (count/min)
- Resting heart rate (count/min)
- Stand time (min)
- Step count (count)
- Walking heart rate average (count/min)

The data incorporated in this analysis is dated from April 17th, 2021 to April 17th, 2024. Some specific dates have been omitted due to lack of data or incompleteness. For verifiability, the datasets are on my GitHub repository:

https://github.com/micahborg/ridge_reg

```
In [1]: import numpy as np
        from sklearn.preprocessing import StandardScaler
        import matplotlib.pyplot as plt
        from sklearn.linear_model import RidgeCV
        from sklearn.model_selection import train_test_split
        from sklearn.metrics import mean_squared_error

        health_data = np.loadtxt('exported_health_data.csv', delimiter=',', skiprows=1, usecols=range(2,11))

        y = health_data[:,0] # this is the response Active energy burned(Cal)
        X = health_data[:,1] # the rest of the features

        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=43)

        # standardize the data
        scaler = StandardScaler()
        scaler.fit(X_train)
        X_train_scaled = scaler.transform(X_train)
        X_test_scaled = scaler.transform(X_test)
```

Model Fitting and Prediction

The dataset was split into a training set and a test set, with 67% of the data used for training and 33% for testing. The features were standardized to have a mean of zero and a standard deviation of one. This ensures that each feature contributes equally to the model's predictions.

```
In [2]: print(np.sum(X_train_scaled, axis=0)) # mean: should be near zero
        print(np.std(X_train_scaled, axis=0)) # standard deviation: should be one
```

```
[-2.65043543e-12 -2.77097789e-13  4.46309656e-14  5.21749310e-13
  5.06620440e-12 -2.72670775e-13  7.50788320e-14 -7.59808882e-14
 -1.81354931e-13]
[1.  1.  1.  1.  1.  1.  1.  1.  1.]
```

A RidgeCV model with built-in cross-validation was employed to find the optimal regularization parameter. Ridge Regression was chosen because health data tends to have similar and/or correlating features, so using a regularization term increases the reliability of the estimates. The model is then used to predict the active energy burned for the test data and a separate dataset containing the most recent week's data.

```
In [3]: alpha_all = np.logspace(-6, 6, 13)
# Fit the Ridge regression model with cross-validation
ridge_model = RidgeCV(alphas=alpha_all, fit_intercept=True, store_cv_values=True)

ridge_model.fit(X_train_scaled, y_train)

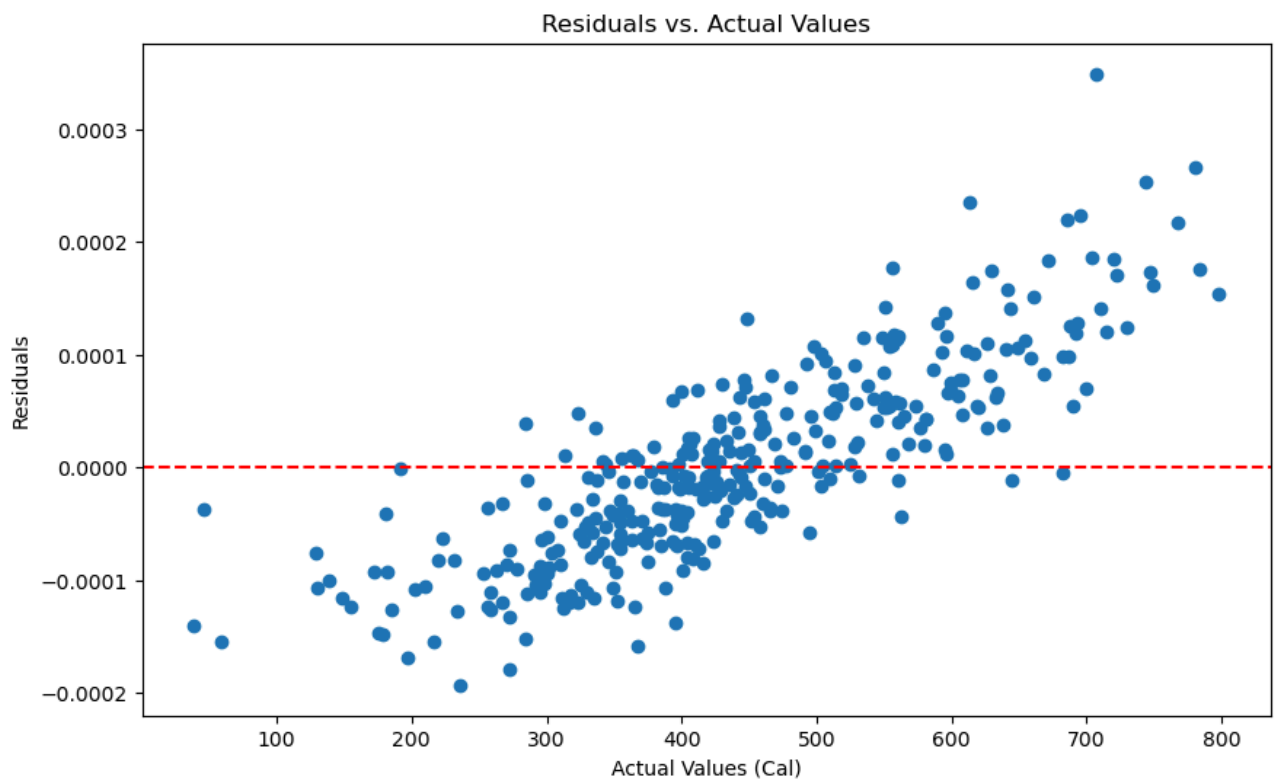
# view model statistics
print('Coefficients:', ridge_model.coef_)
print('Alpha:', ridge_model.alpha_)
print('Score:', ridge_model.score(X_train_scaled, y_train))

Coefficients: [ 1.47244089e+02 -2.13819635e-05 -1.02634383e-05 -5.02420450e-06
 -1.60778450e-05 -7.60674701e-06 -1.70051419e-05 -2.47931559e-05
 -1.62376220e-05]
Alpha: 1e-06
Score: 0.9999999999996199
```

```
In [4]: # predict and evaluate the model
y_pred = ridge_model.predict(X_test_scaled)
mse = mean_squared_error(y_test, y_pred)
print('MSE:', mse)

MSE: 7.433600953388366e-09
```

```
In [5]: residuals = y_test - y_pred
plt.figure(figsize=(10, 6))
plt.scatter(y_test, residuals)
plt.title('Residuals vs. Actual Values')
plt.xlabel('Actual Values (Cal)')
plt.ylabel('Residuals')
plt.axhline(y=0, color='r', linestyle='--')
plt.show()
```



Interpretation of Prediction Results

A few outliers suggest some cases where the model's predictions were off. However, the differences are incredibly small (-0.0002 and 0.0003, respectively). Still, the residual analysis showed that the model predicts the energy expenditure close to accurately for most observations.

```
In [6]: new_week_data = np.loadtxt('this_week_data.csv', delimiter=',', skiprows=1, usecols=range(2,11))

y_new = new_week_data[:,0] # this is the response Active energy burned(Cal)
X_new = new_week_data[:,1] # the rest of the features

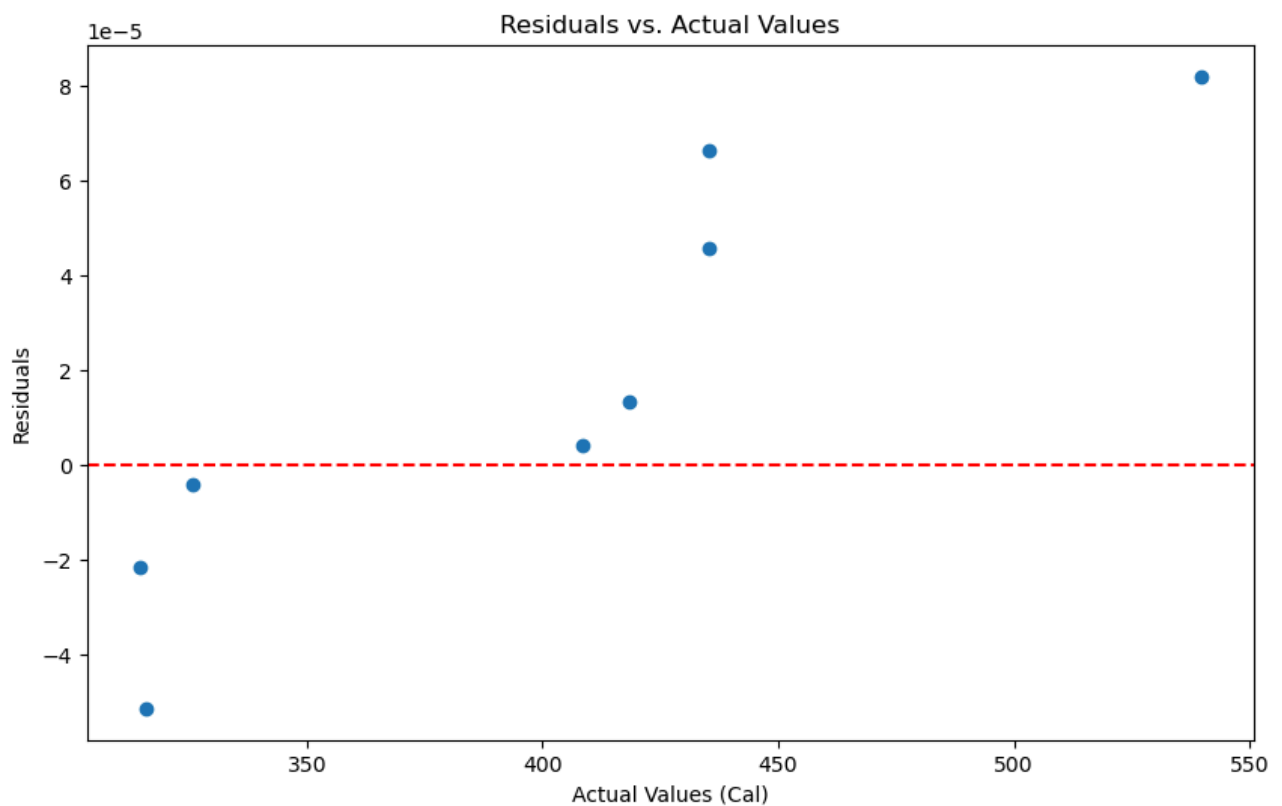
X_new_scaled = scaler.transform(X_new)

y_new_pred = ridge_model.predict(X_new_scaled)

print('Predicted values:', y_new_pred)

residuals = y_new - y_new_pred
plt.figure(figsize=(10, 6))
plt.scatter(y_new, residuals)
plt.title('Residuals vs. Actual Values')
plt.xlabel('Actual Values (Cal)')
plt.ylabel('Residuals')
plt.axhline(y=0, color='r', linestyle='--')
plt.show()
```

Predicted values: [418.38698667 435.36995413 539.52391789 315.96605139 435.32193354
314.62602161 408.49299583 325.78800421]



Conclusion

The residuals for the current week's health data is higher than the original test data, but only by up to 8 calories in one prediction. For the future, different features could be selected to find further prediction measurements. The model has demonstrated the capacity to "reasonably" predict daily active energy expenditure based on the selected health metrics. The majority of the model's predictions closely match the actual data points and predict nearly the desired active calories burned for the day according to my Apple Watch.