Linear Regression

Ishita Jaju
16BCE1059

1 The algorithm

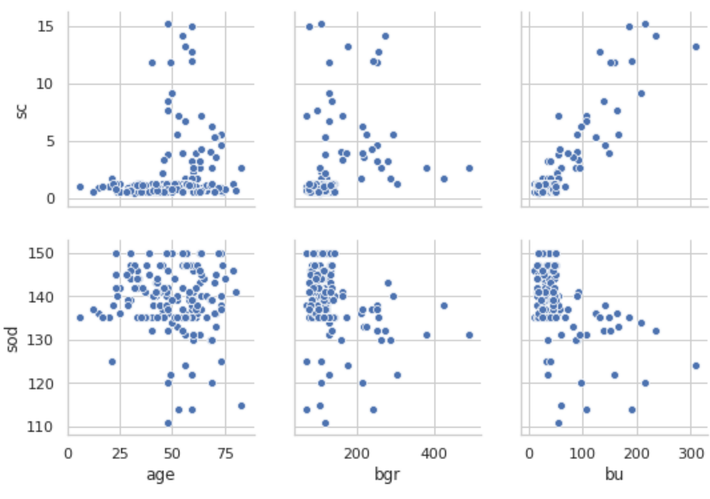
Linear regression works on finding a hypothesis $h_\theta(x)$ with weights in the form of weight vector θ to minimize the cost function J. The hypothesis function is in the form of a linear function

$$h_\theta(x) = \theta^T X$$

where $\theta^T X = \theta_0 + \theta_1 x_1 + \theta_2 x_2 ...$
Here, θ is the weight vector and X is the input feature matrix.

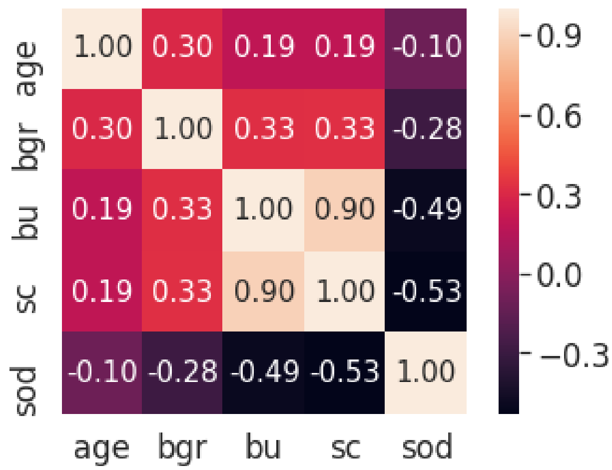
2 Testing

After running a simple pairplot:



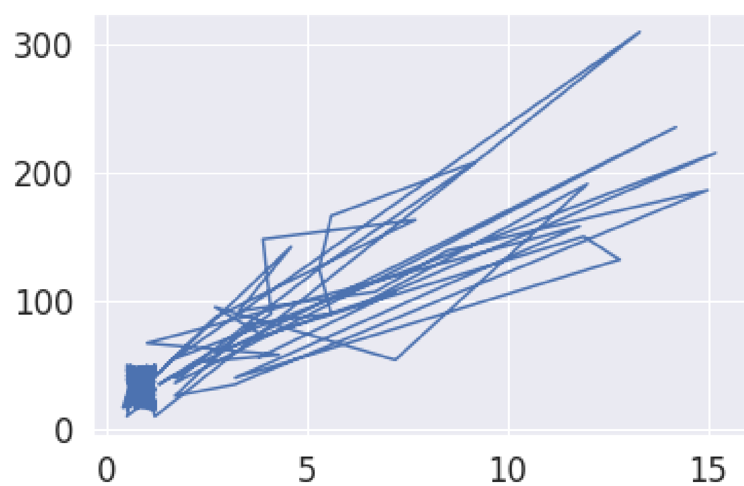
Our best bet for a good inear fit would be to go for *sc* vs *bu*

Its correlation matrix also gives us the maximum value for the same.




```
X = df[['sc']].values
y = df['bu'].values

plt.plot(X,y)
plt.show()
```



3 Implentation

3.1 Using Gradient Descent

Here, the cost function to be minimized is

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)} - y^{(i)})^2$$

which is based on the simple least squares method. The gradient descent algorithm goes like this:

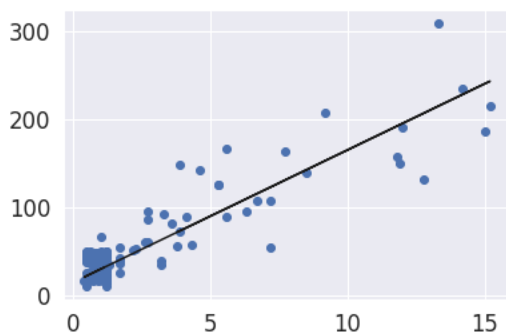
Repeat until convergence $\{$

$$\theta_j \leftarrow \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta)$$

$\}$

On setting $\alpha = 0.001$ and with 1500 iterations, we get the regression line

```
iterations= 1500
theta0 = [14.93262201] theta1 = [15.0012666]
slope= 15.001266603540925 intercept= 20.933128650320924
```



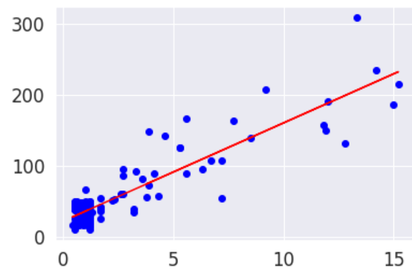
3.2 Using the sklearn library

```
from sklearn.linear_model import LinearRegression
slr = LinearRegression()
slr.fit(X, y)
print('Slope: %.3f' % slr.coef_[0])

print('Intercept: %.3f' % slr.intercept_)
```

```
Slope: 13.800
Intercept: 22.373
```

```
lin_regplot(X, y, slr)
plt.show()
```

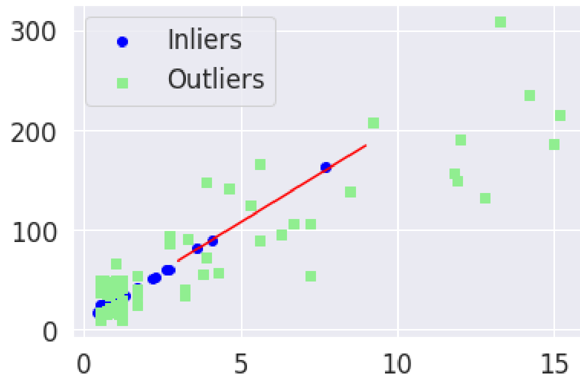


3.3 Using RANSAC

RANSAC (Random sample consensus) is an iterative method to estimate parameters of a mathematical model from a set of observed data that contains outliers, when outliers are to be accorded no influence on the values of the estimates. Therefore, it also can be interpreted as an outlier detection method. RANSAC uses the voting scheme to find the optimal fitting result.

1. In the first step, a sample subset containing minimal data items is randomly selected from the input dataset. A fitting model and the corresponding model parameters are computed using only the elements of this sample subset.

2. In the second step, the algorithm checks which elements of the entire dataset are consistent with the model instantiated by the estimated model parameters obtained from the first step.




```
print('Slope: %.3f' % ransac.estimator_.coef_[0])

print('Intercept: %.3f' % ransac.estimator_.intercept_)
```

```
Slope: 19.272
Intercept: 11.483
```

4 Comparisons and inference

1. For the Gradient Descent model,
   Slope = 15.001266603540925
   Intercept = 20.933128650320924

2. For the sklearn Regression model,
   Slope = 13.800
   Intercept = 22.373

3. For the RANSAC model,
   Slope = 19.272
   Intercept = 11.483

Hence, the slopes and intercepts are slightly similar.