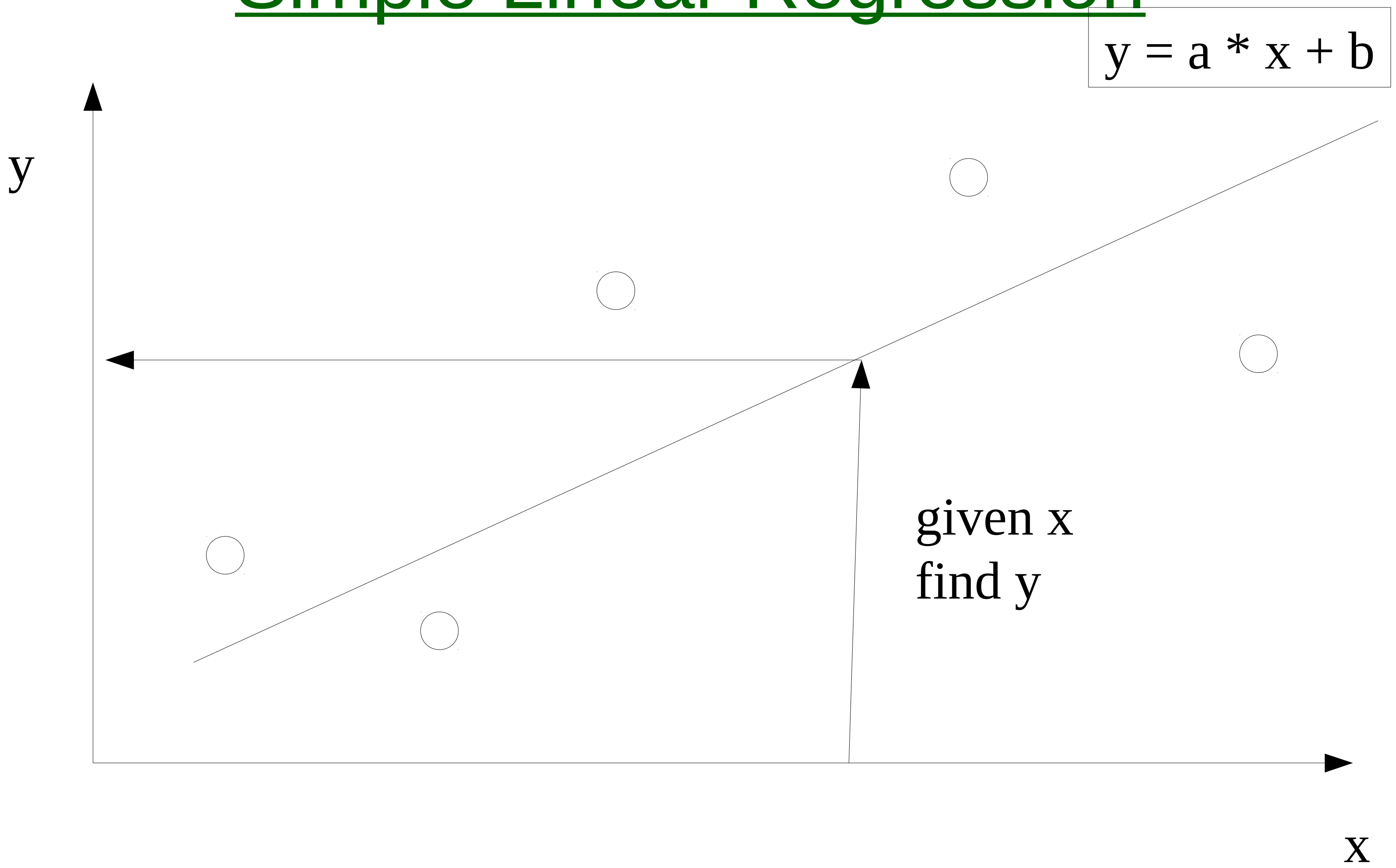
Simple Linear Regression

Simple Linear Regression

- Called simple because the is only one independent variable (x).
- The dependent or target variable is y.
- Given x we want to predict y.
- Simple LR is covered for demonstration purposes only.
- In practice it is rarely used.
- A simple linear regression example is not a suitable example for an assignment.

Simple Linear Regression



Simple Linear Regression

- The least squares method produces a straight line that minimizes the sum of the squares of the errors.
- Errors (or Residuals)
  - differences between predicted value of y and actual value of y.
  - $\hat{y} - y$
- The objective function is the sum of the squares of the errors (SSE).

Solution

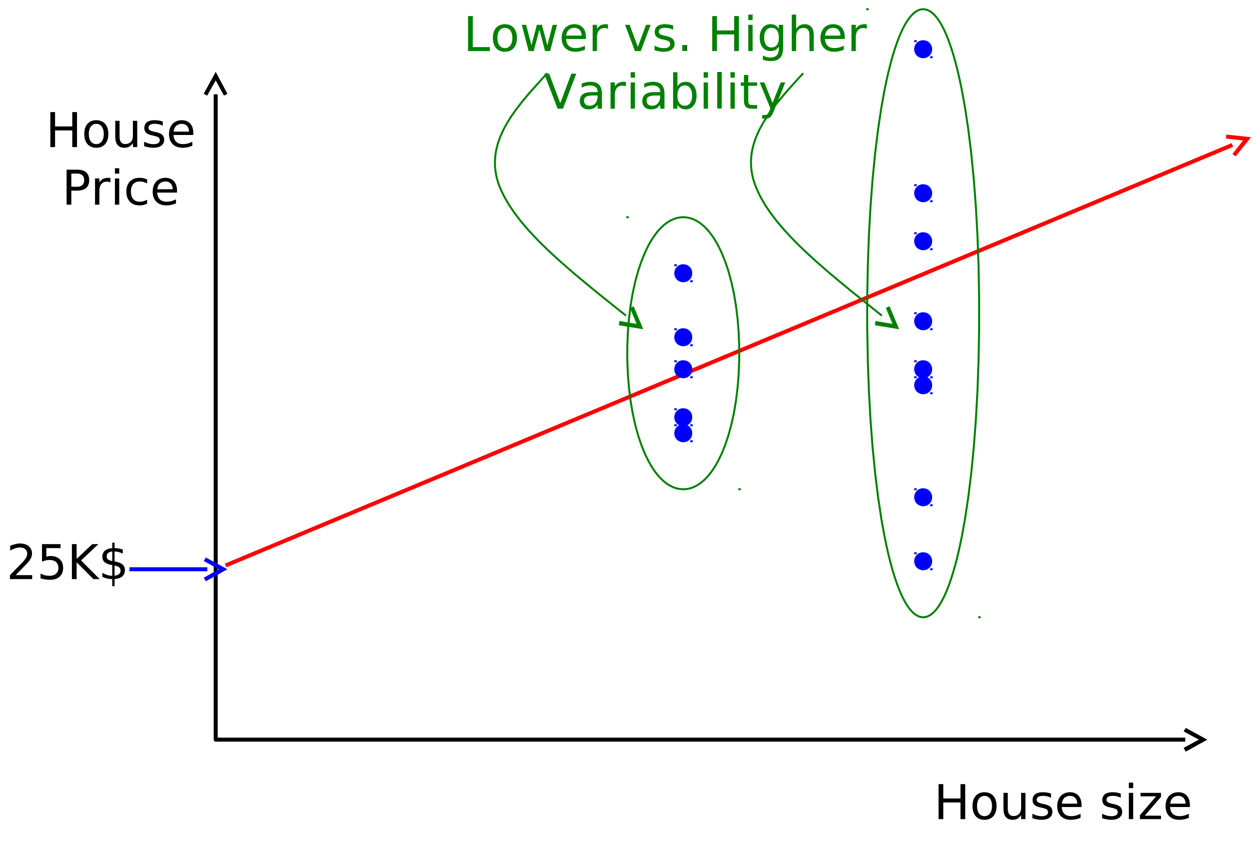
- $y = ax + b$
- Minimising SSE gives us formulas for a and b.
  - $a = (n\Sigma xy - (\Sigma x)(\Sigma y)) / (n\Sigma x^2 - (\Sigma x)^2)$
  - $b = (\Sigma y - a(\Sigma x)) / n$
- Very simple to calculate

Conditions Necessary for Linear Regression Analysis

- The probability distribution of errors is normal.
- The mean of the errors is 0
- The standard deviation of the errors is a constant regardless of the value of x.
- The value of the error associated with any particular value of y is independent of the error associated with any other value of y.

Counter Example
sigma not constant (3 above)

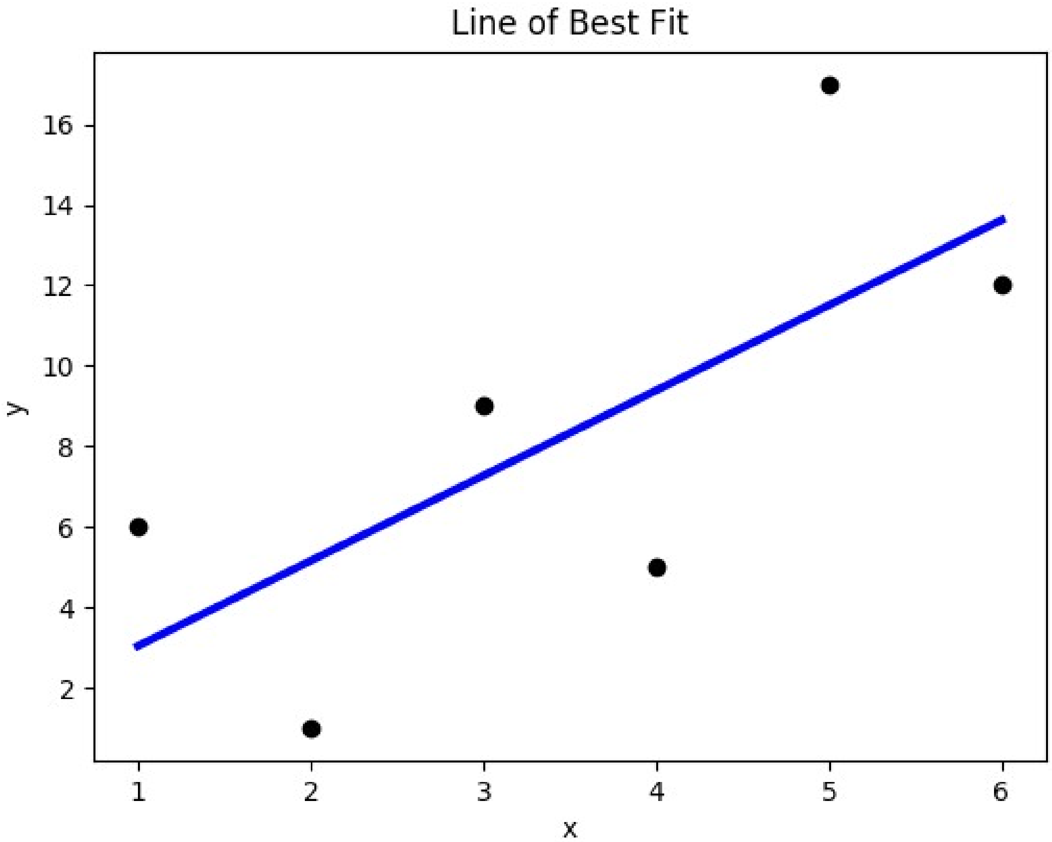
House Price = 25,000 + 75(Size)



Simple Linear Regression in Python

```
import numpy as np
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt

x = np.array([1, 2, 3, 4, 5, 6]).reshape(-1, 1)
y = np.array([6,1,9,5,17,12])
print("x.shape ", x.shape)

model = LinearRegression()
model.fit(x, y)
print('intercept:', model.intercept_)
print('slope:', model.coef_)

y_hat = model.predict(x)
print('predicted response:', y_hat, sep='\n')

plt.scatter(x, y,  color='black')
plt.plot(x, y_hat, color='blue', linewidth=3)
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

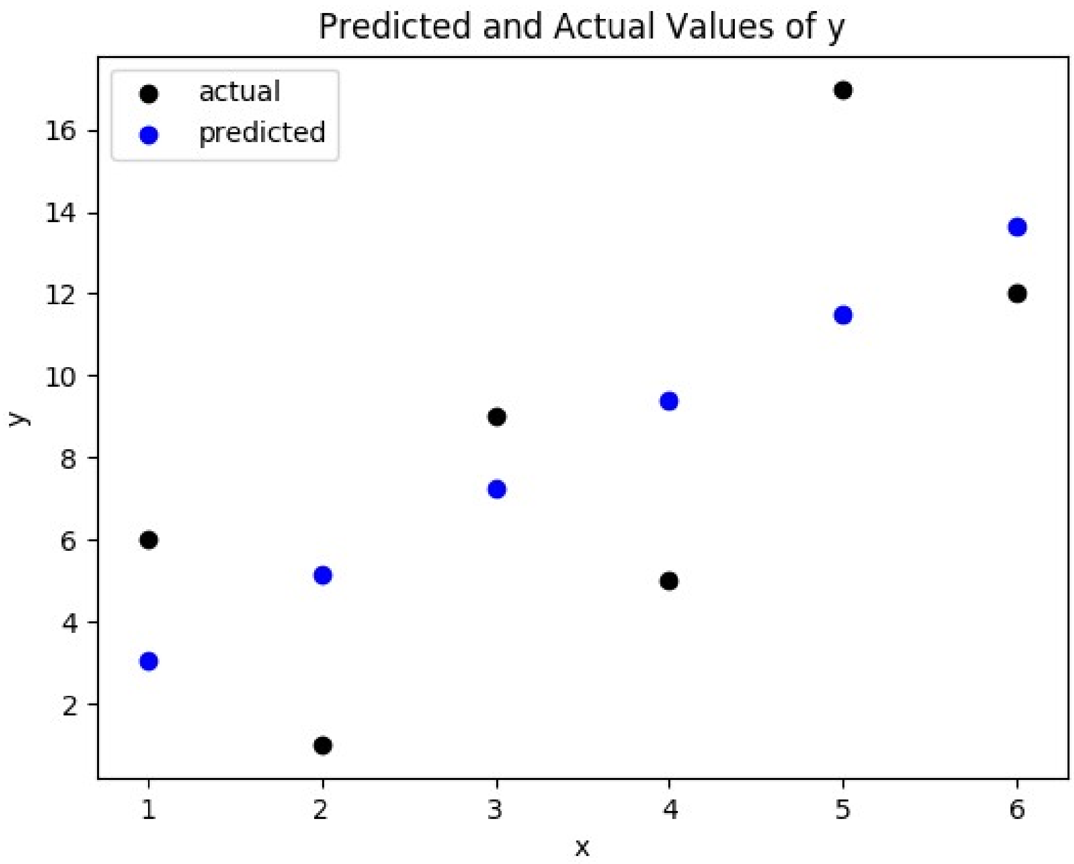
Line of Best Fit



Prediction

```
plt.scatter(x, y,  color='black', label='actual')
plt.scatter(x, y_hat, color='blue', label='predicted')
plt.xlabel('x')
plt.ylabel('y')
plt.legend(loc='upper left')
plt.title('Predicted and Actual Values of y')
# plt.show()
plt.savefig('plots/p2predictedActual.png')

# Find RMSE
yhat = model.predict(X_test)
print(mean_squared_error(y_test, yhat, squared=False))
```

Predicted and Actual Values of y



Linear Regression in Python

- x is of shape (6,1)
  - 2D array, 6 rows, 1 column
- y is of shape (6,)
  - 1D array
- model = LinearRegression()
- model.fit(x, y)
- y_hat = model.predict(x)