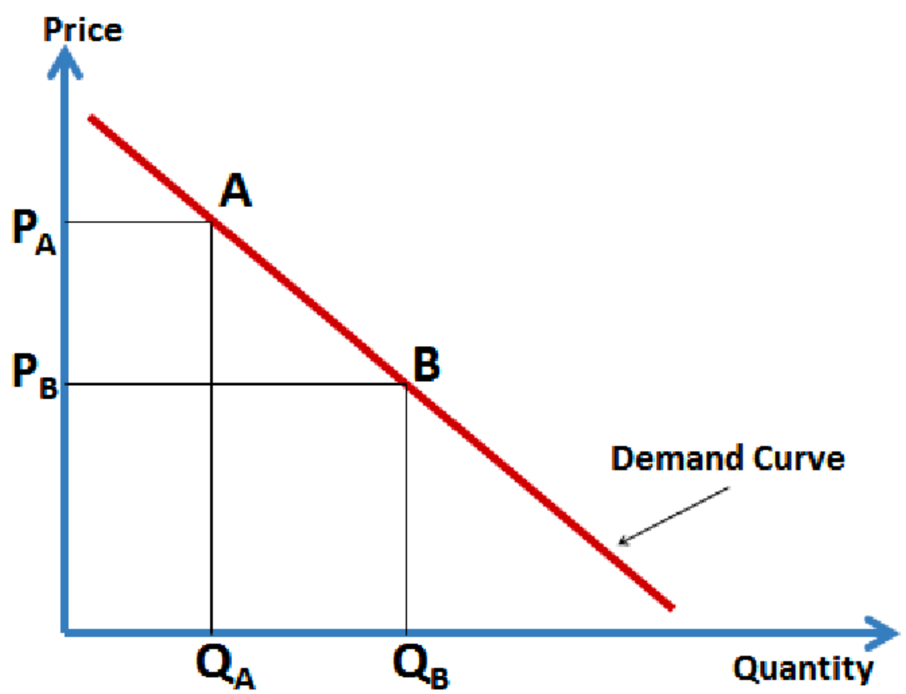
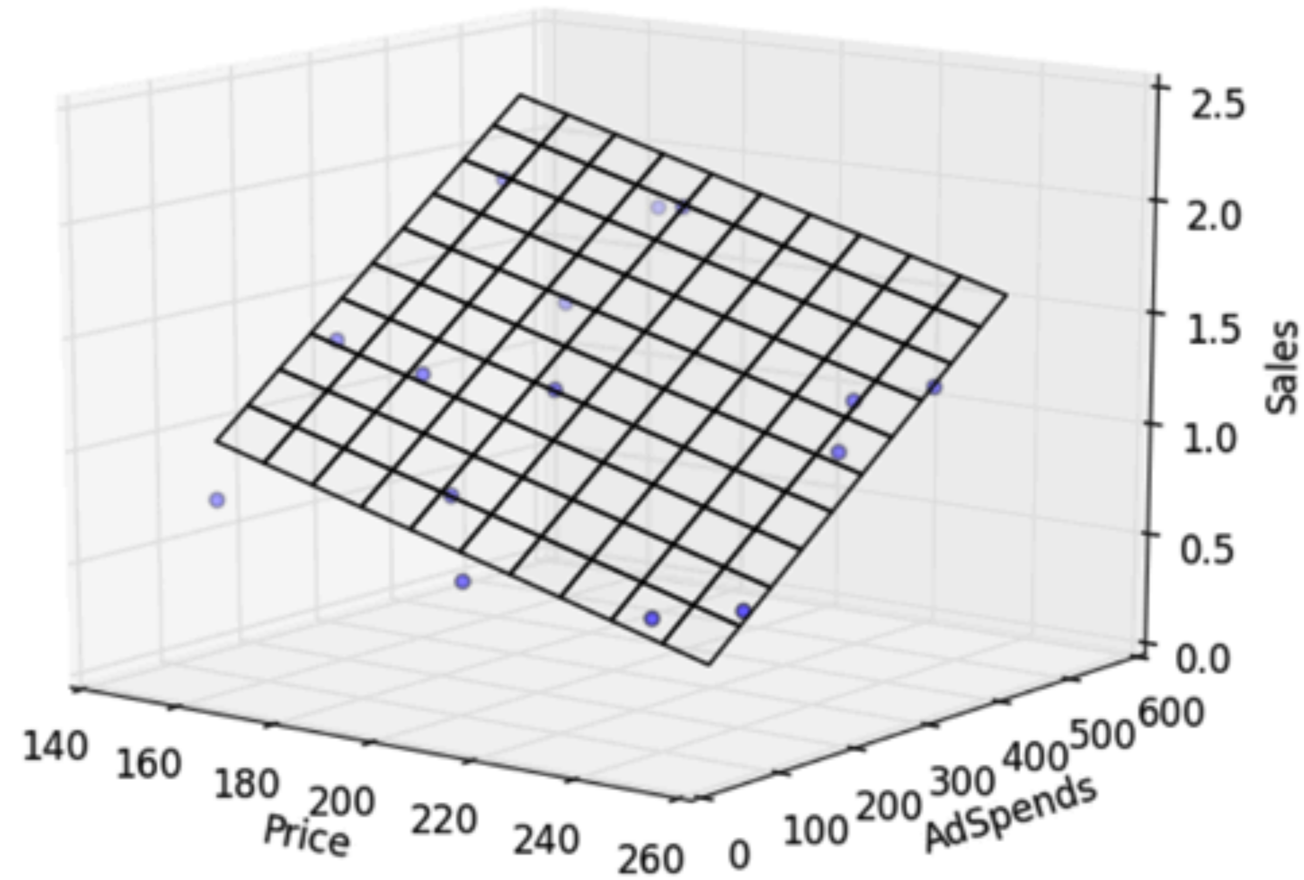
Week 5: Data Science Part-Time Course

Linear Regression

Dami Lasisi

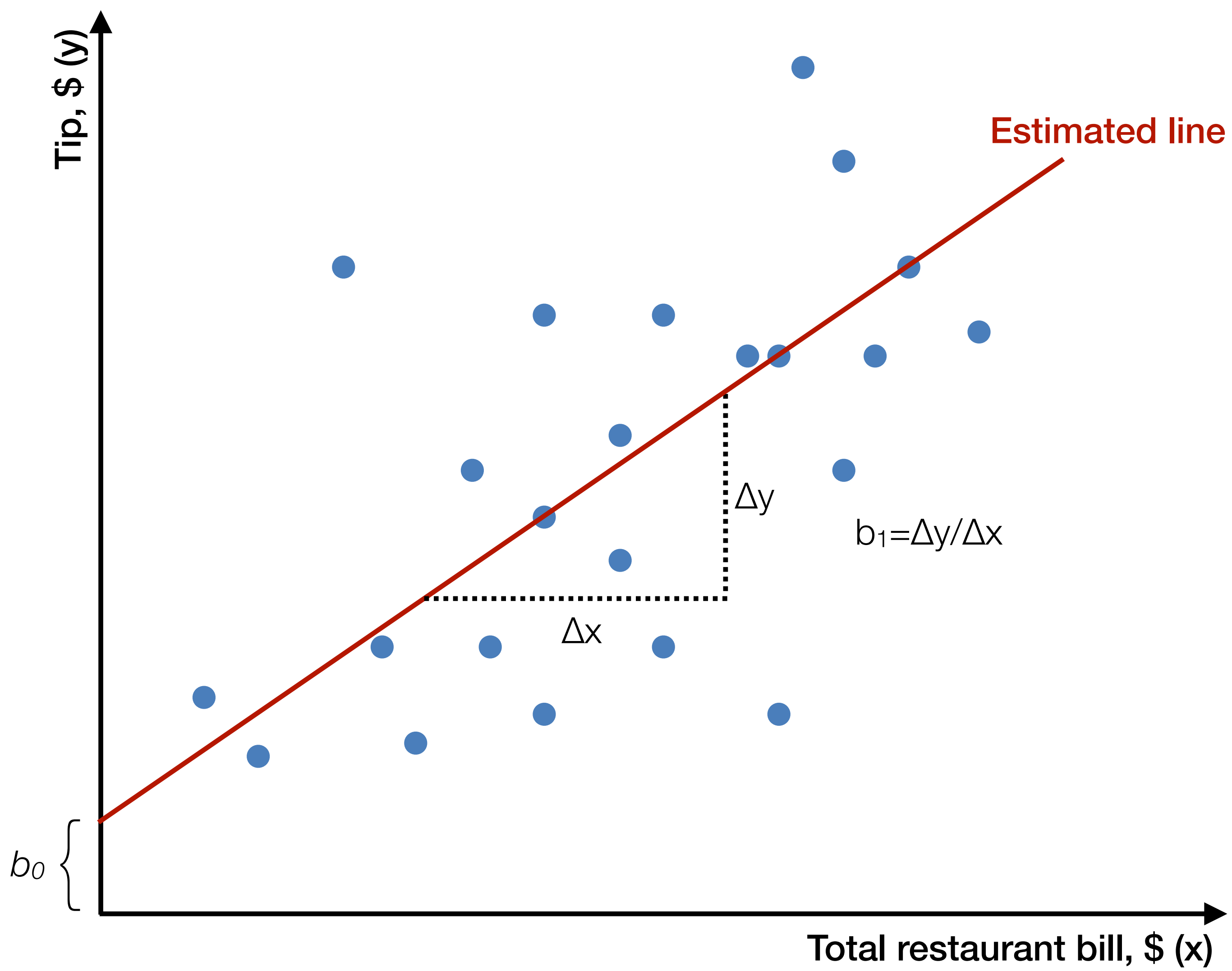


Source: https://www.researchgate.net/figure/The-Conventional-Demand-Curve_fig1_262309137



Source: https://medium.com/@dhwajraj/python-regression-analysis-part-4-multiple-linear-regression-ed09a0c31c74

We use linear regression to find statistical linear relationships between a target (dependent variable) and one or more features (predictors or independent variables).



Equation of a straight line:

$y = mx + b$

Linear Regression Equation:

$y = b_0 + b_1 X_1$

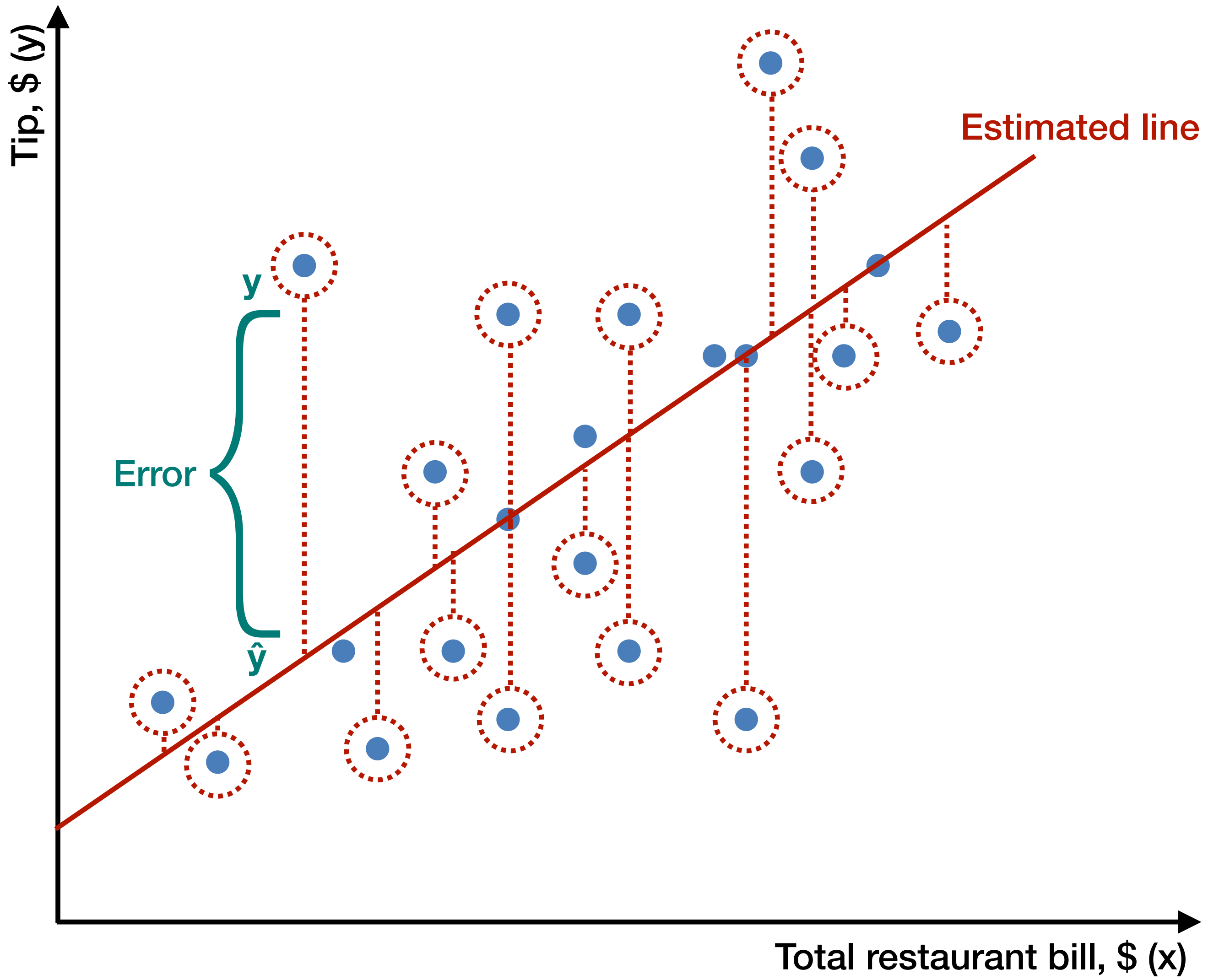
$Tip = b_0 + b_1 \text{Total restaurant bill}$

Multiple Linear Regression Equation:

$y = b_0 + b_1 X_1 + b_2 X_2 + \ldots + b_n X_n$

$Tip = b_0 + b_1 \text{Total restaurant bill} + b_2 \text{Waiter's attitude} + \ldots + b_n \text{Party Size}$

b_0: Intercept
$b_1, \ldots, b_n$: Coefficients
$X_1, \ldots, X_n$: Independent variables (features)
y: Dependent variable (target)

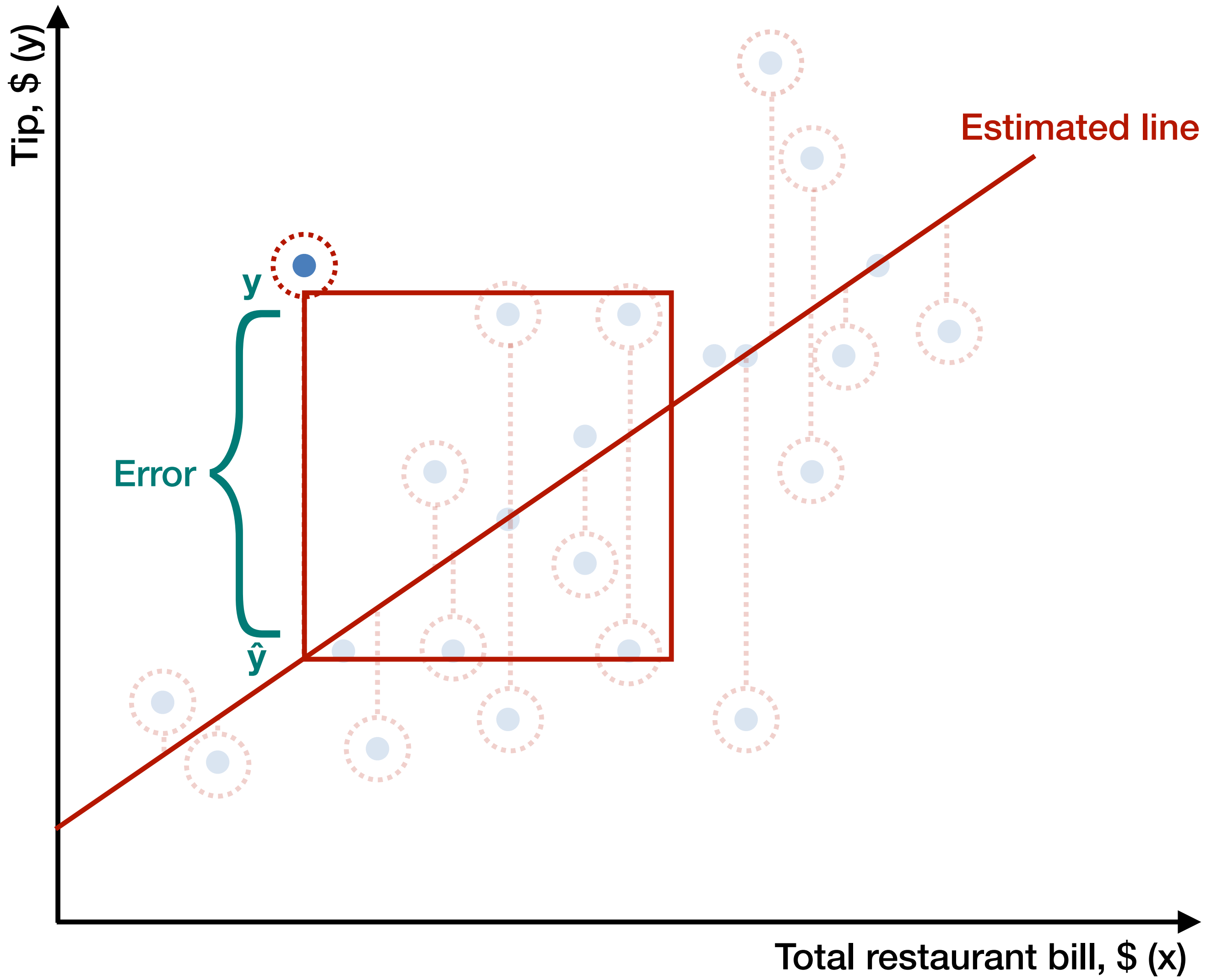


Find the best fit line that has the overall minimum error (i.e the distance between the predicted output and the actual output)

$\hat{y} = b_0 + b_1 X_1$
$y = \hat{y} + Error$
$Error = y - \hat{y}$

Objective: find that best b_0 and b_1 that minimizes $\sum(y - \hat{y})^2$

SSE/SSR/RSS



Find the best fit line that has the overall minimum error (i.e the distance between the predicted output and the actual output)

$\hat{y} = b_0 + b_1 X_1$
$y = \hat{y} + Error$
$Error = y - \hat{y}$

Objective: find that best b_0 and b_1 that minimizes $\sum(y - \hat{y})^2$

SSE/SSR/RSS

Assumptions of Linear Regression

- Linear and additive relationship between dependent and independent variables
- No autocorrelation
- No multicollinearity
- No heteroskedasticity
- Error terms are normally distributed

Building a Linear Regression Model in Python

Building a Linear Regression Model in Python: Scikit Learn

In [ ]:
```
from sklearn.linear_model import LinearRegression
```

In [ ]:
```
linreg = LinearRegression()
linreg.fit(X, y)
```

In [ ]:
```
print(linreg.intercept_)
print(linreg.coef_)
```

In [ ]:
```
linreg.predict(77)
```

Building a Linear Regression Model in Python: Statsmodel

```
In [ ]: import statsmodels.regression.linear_model as sm
```

```
In [ ]: lr = sm.OLS(y,X)
        results = lr.fit()
        print(results.params)
```

```
In [ ]: result.summary()
```

Interpreting Simple Linear Regression Models

Assume that for our restaurant tips example, the model is:

$$Tip = 0.12 + 0.67\,Total\ restaurant\ bill$$

This means that for every dollar increase in total restaurant bill, a waiter's tip will increase on average by 67 cents. When the restaurant bill is $0, tip will be 12 cents on average.

Interpreting Multiple Linear Regression Models

Assume that for our restaurant tips example, the model is:

Tip = 0.08 + 0.54Total restaurant bill + 0.98Waiter's attitude - 0.07Party Size

Holding all other variables constant, for every dollar increase in total restaurant bill, a waiter's tip will increase on average by 54 cents.

Holding all other variables constant, for every value increase in a waiter's attitude, the waiter's tip will increase on average by 98 cents.

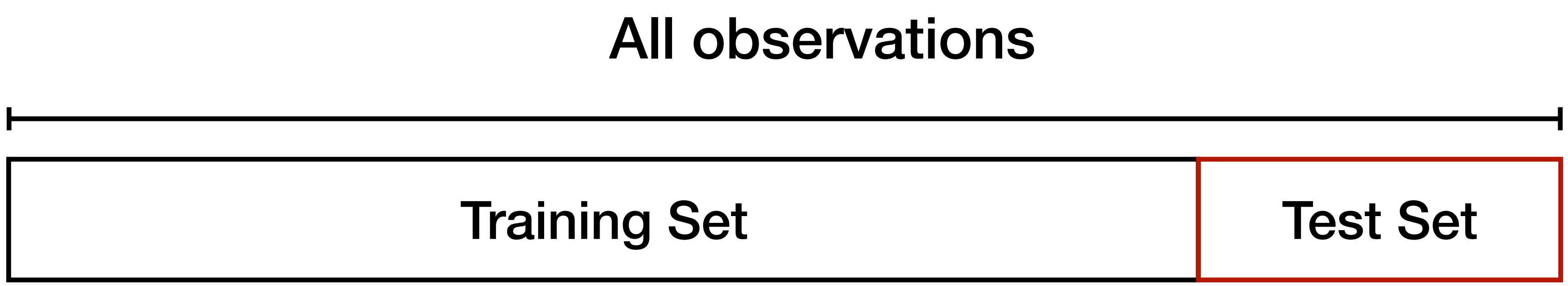
Holding all other variables constant, for every additional person in party size, a waiter's tip will decrease on average by 7 cents.

When the restaurant bill is $0, tip will be 8 cents on average.

Evaluation Metrics for Regression

- **Mean absolute error (MAE):** mean of the absolute values of the errors
- **Mean squared error (MSE):** mean of the squared errors
- **Root mean squared error (RMSE):** square root of the mean of the squared errors

Train/Test Split in Python




```
In [ ]: from sklearn import metrics
        from sklearn.model_selection import train_test_split
```

```
In [ ]: X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.7, random_state=123)

        linreg = LinearRegression()
        linreg.fit(X_train, y_train)

        y_pred = linreg.predict(X_test)
        print("MSE:", np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```