Relationships Between Variables Cheat Sheet

Hamed Ahmadinia, Ph.D.
Hamed.Ahmadinia@metropolia.fi

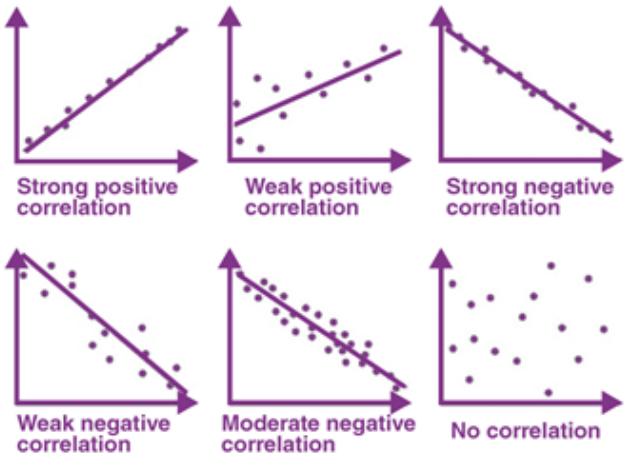
Covariance

Definition: Covariance shows how two variables move together.
Types:

- **Positive**: Both variables increase or decrease together.
- **Negative**: One variable increases, the other decreases.
- **Zero**: No relationship.

Calculate Covariance in Python:

```
import numpy as np
cov_matrix = np.cov(x, y)
```



Correlation

Definition: Correlation measures the strength and direction of a relationship.
Range of Correlation Coefficient (r):

- **+1**: Perfect positive correlation.
- **0**: No correlation.
- **-1**: Perfect negative correlation.

Calculate Correlation in Python:

```
import numpy as np
corr_coeff = np.corrcoef(x, y)[0,1]
```

Types of Correlation Metrics

Pearson: Linear relationships (sensitive to outliers).
Spearman: Works with ranked data and non-linear relationships.
Kendall's Tau: Preferred for small ordinal datasets.

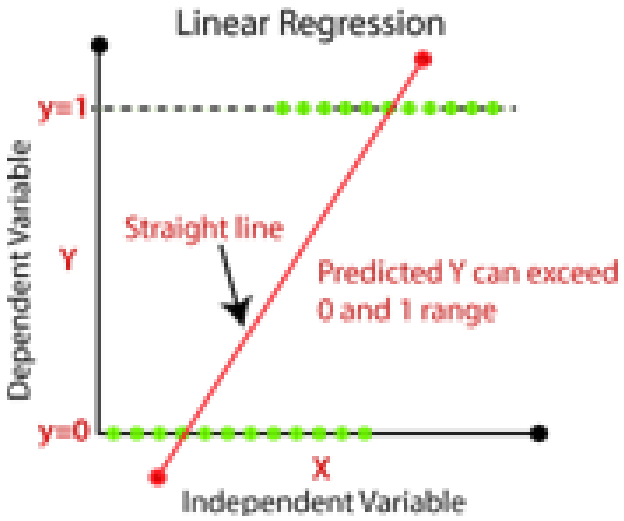
Linear Regression

Key Concepts:

- **Intercept**: Baseline prediction without features.
- **Coefficients**: Impact of input variables.
- **R-squared (R^2)**: Model fit quality (closer to 1 is better).
- **MSE (Mean Squared Error)**: Measures prediction accuracy (lower is better).

Linear Regression in Python:

```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(X, y)
intercept = model.intercept_
coefficients = model.coef_
```



Evaluating Model Performance

Underfitting: Too simple, missing patterns.
Proper Fit: Captures general trends correctly.
Overfitting: Too complex, fitting noise instead of data.

Loss Function in Python:

```
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(y_true, y_pred)
```

Best Practices

- Use **appropriate correlation metrics** based on data type.
- Apply **feature scaling** before regression.
- Evaluate model fit using **R-squared** and **MSE**.
- Ensure training/testing split for validation.

Website: ahmadinia.fi