Supervised Learning Session-2

SUMMARY of PREVIOUS CLASS

INTRODUCTION

- What is Machine Learning?
- Types of ML
- Input-Features-Independent Variables
- Target-Desired Output-Dependent Variables-Labels
- Observations-Samples

LINEAR REGRESSION

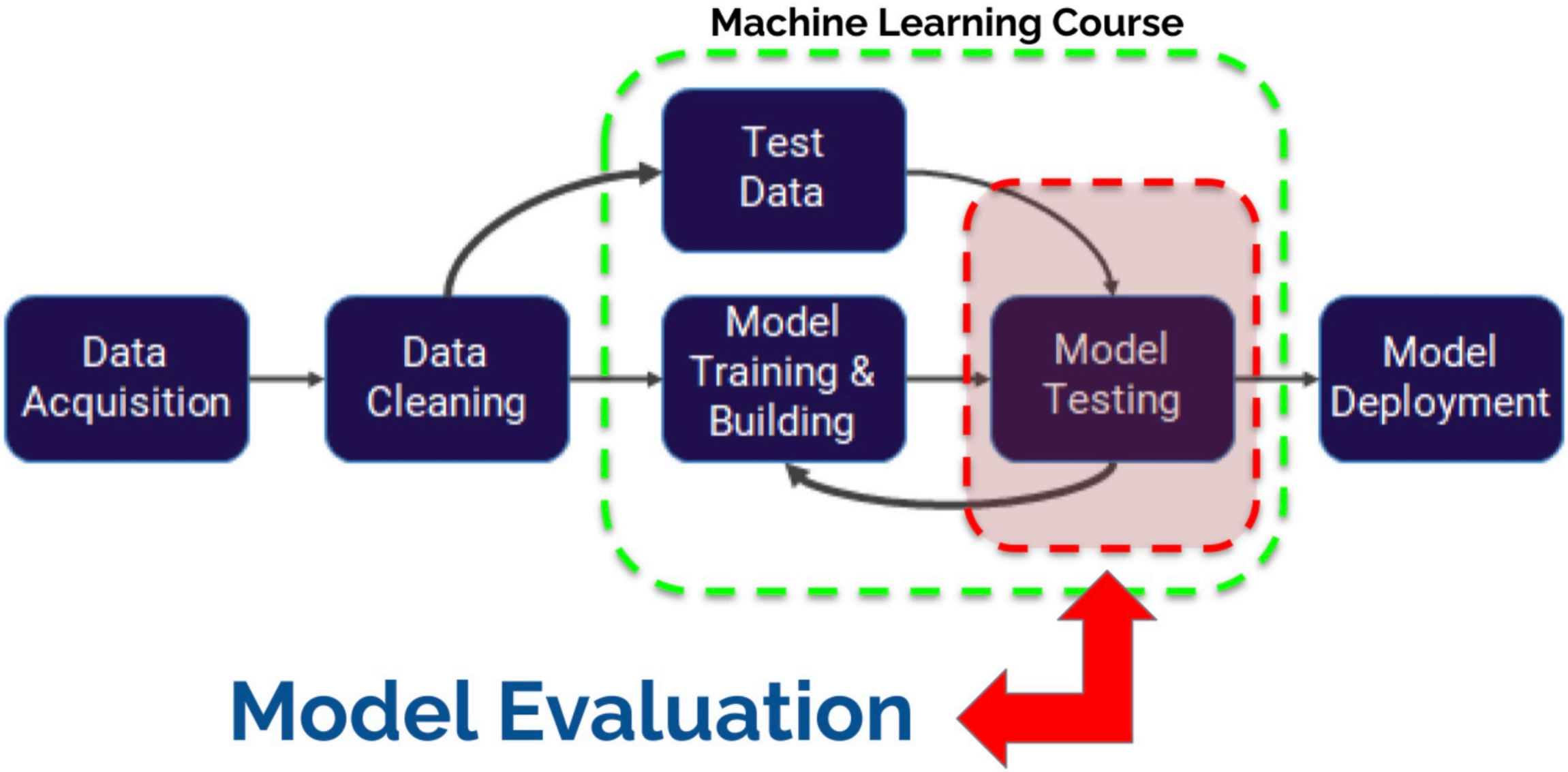
- Why do we need linear regression?
- Terms of Correlation & Regression
- Simple & Multiple Linear Regression
- Residue minimization techniques (LSE, Gradient Descent)

- Evaluation Metric: R_Square(R^2) $R^2 = 1 - \frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2}$

- Residuals
- Regression Error Metrics
- Scikit-learn Library

Session-2

Where are we?



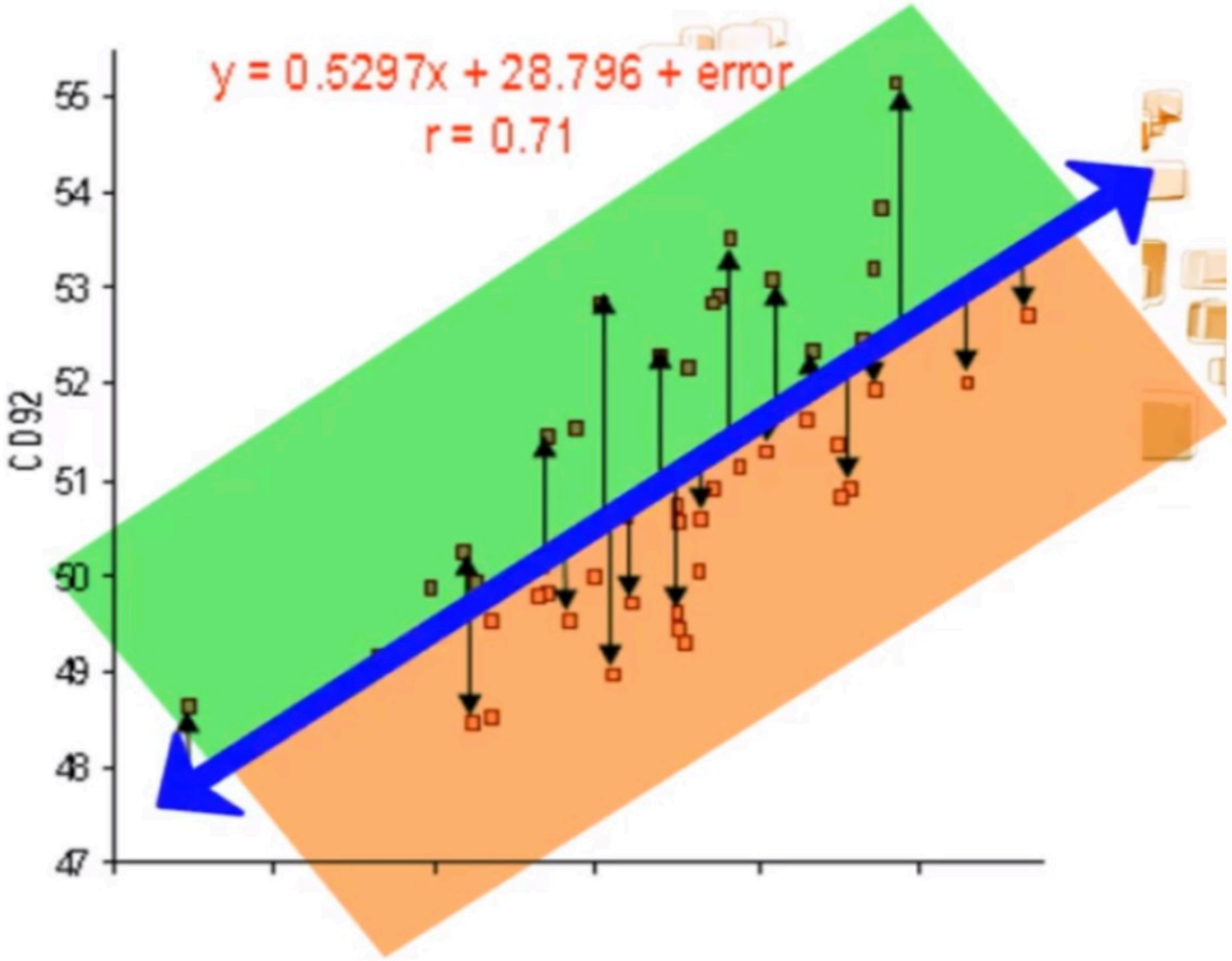
Model Evaluation

Residuals

Residuals

What is a Residual in Regression?

- When you perform regression, you get a line of best fit.
- A residual is the vertical distance between a data point and the regression line.
- Each data point has one residual:
  - Positive: above the regression line.
  - Negative: below the line.
  - Zero: the line passes through the point.



picture credit: What are Residuals in Regression? (Prof. Essa)

Residuals

Errors

- Residuals are sometimes called "**errors**."
- It means that there is some unexplained difference – an error that isn't explained by the regression line.

picture credit: What are Residuals in Regression? (Prof. Essa)

Residuals

Equation

- The residual(*e*) can also be expressed with an **equation**.
- The *e* is the difference between the predicted value ($\hat{y}$) and the observed value
- **Residual = Observed value – predicted value: $e = y - \hat{y}$**

Sum and Mean of Residuals Always Equal to Zero

picture credit: What are Residuals in Regression? (Prof. Essa)

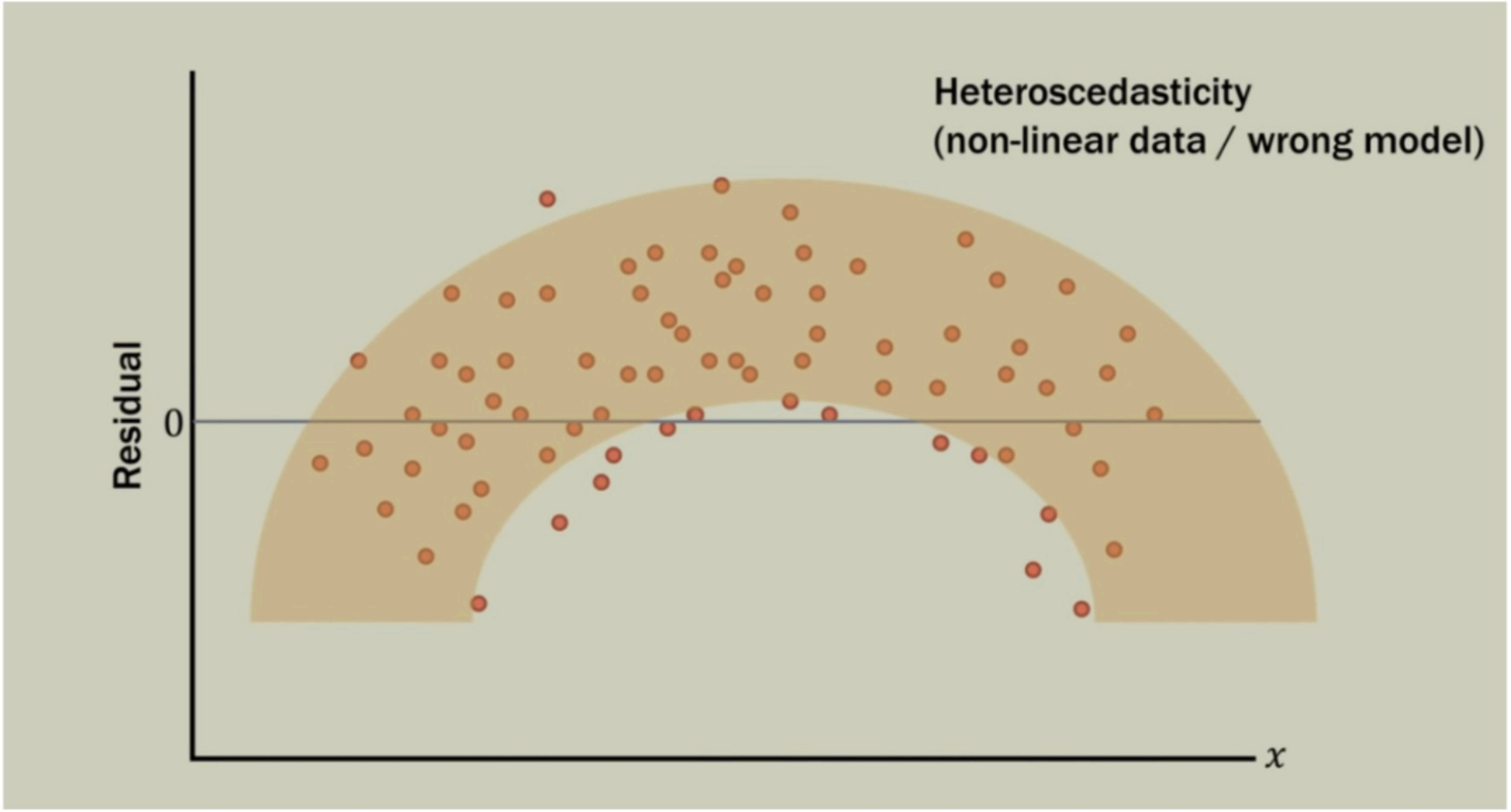
Residuals

A few characteristics of a good residual plot are as follows:

- The most important assumption of a linear regression model is that the ***errors are independent and normally distributed.***
- It has a high density of points close to the origin and a low density of points away from the origin
- It is symmetric about the origin

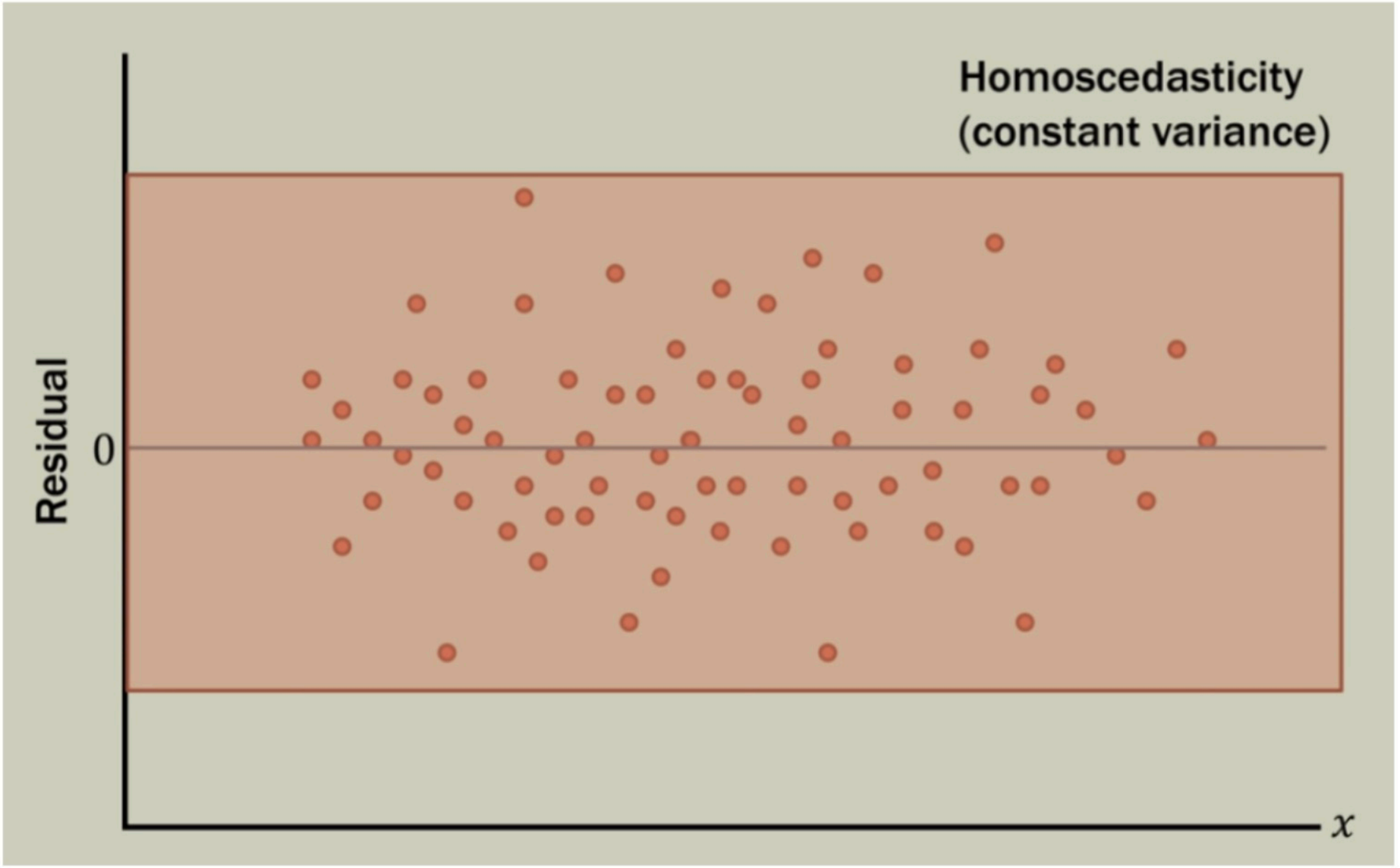
picture credit: Statistics 101: Linear Regression, Residual Analysis (Brandon Foltz)

Residuals



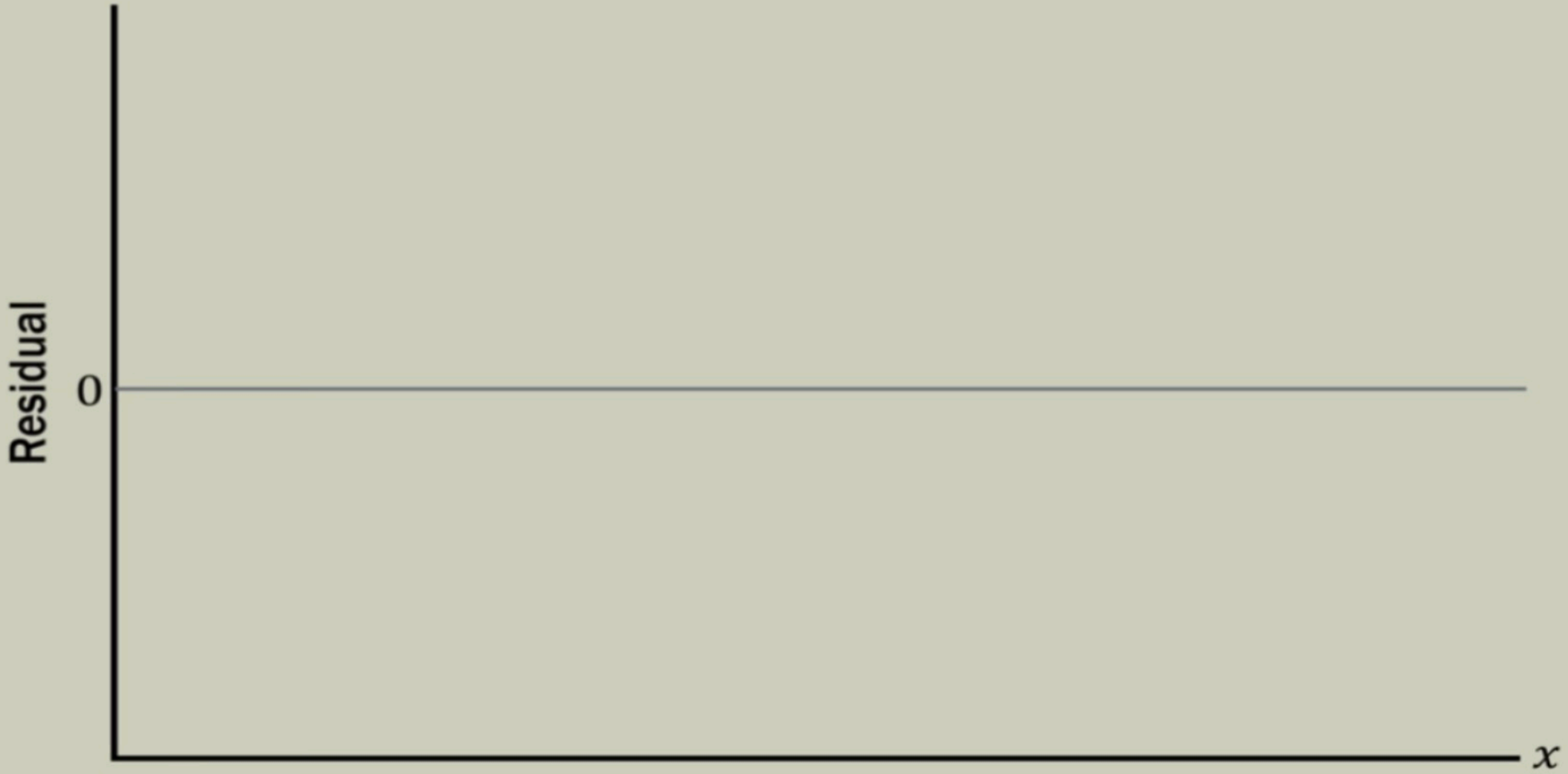
picture credit: Statistics 101: Linear Regression, Residual Analysis (Brandon Foltz)

Residuals



picture credit: Statistics 101: Linear Regression, Residual Analysis (Brandon Foltz)

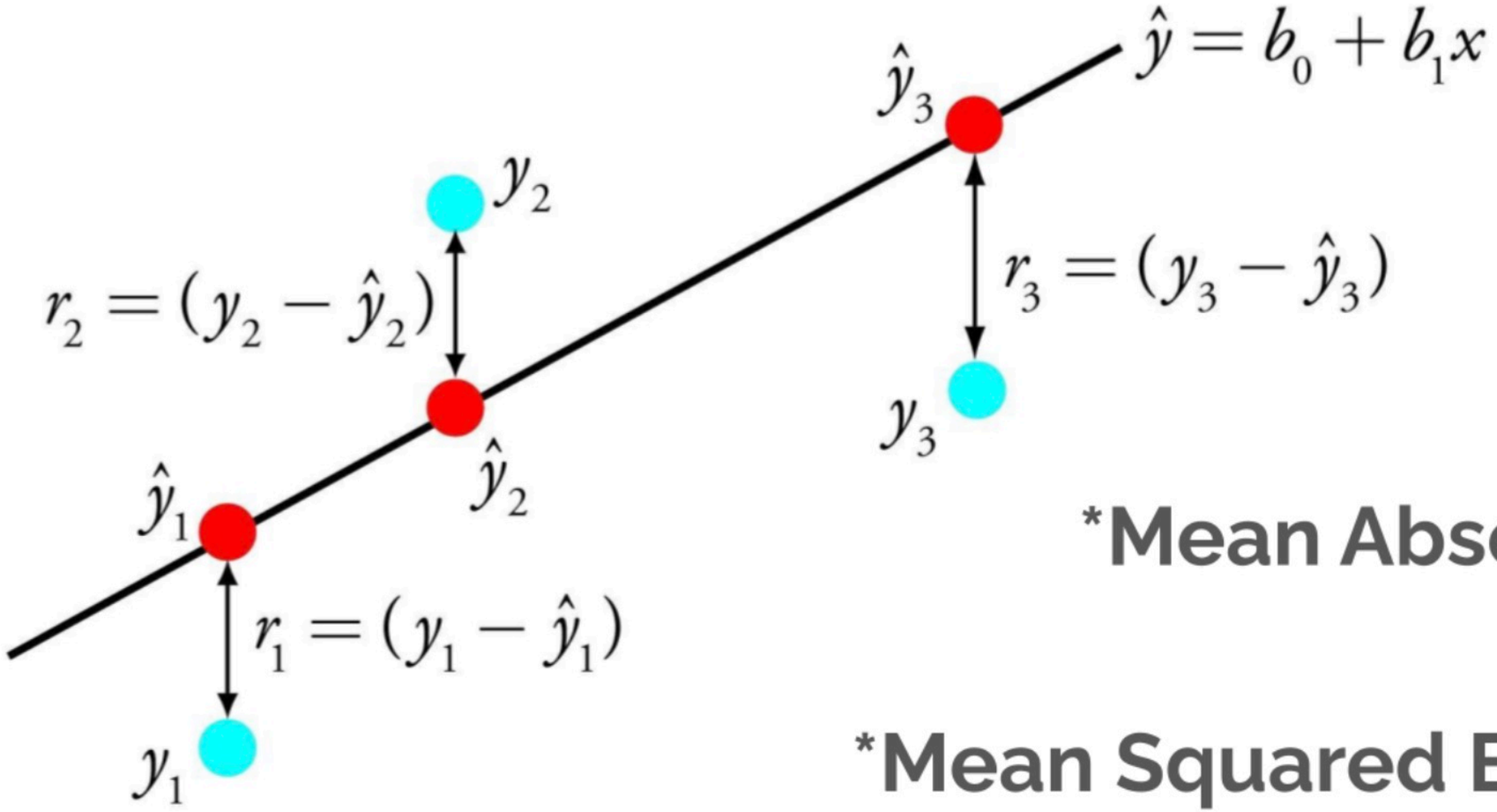
Residuals



picture credit: Statistics 101: Linear Regression, Residual Analysis (Brandon Foltz)

Regression Error Metrics

Regression Error Metrics

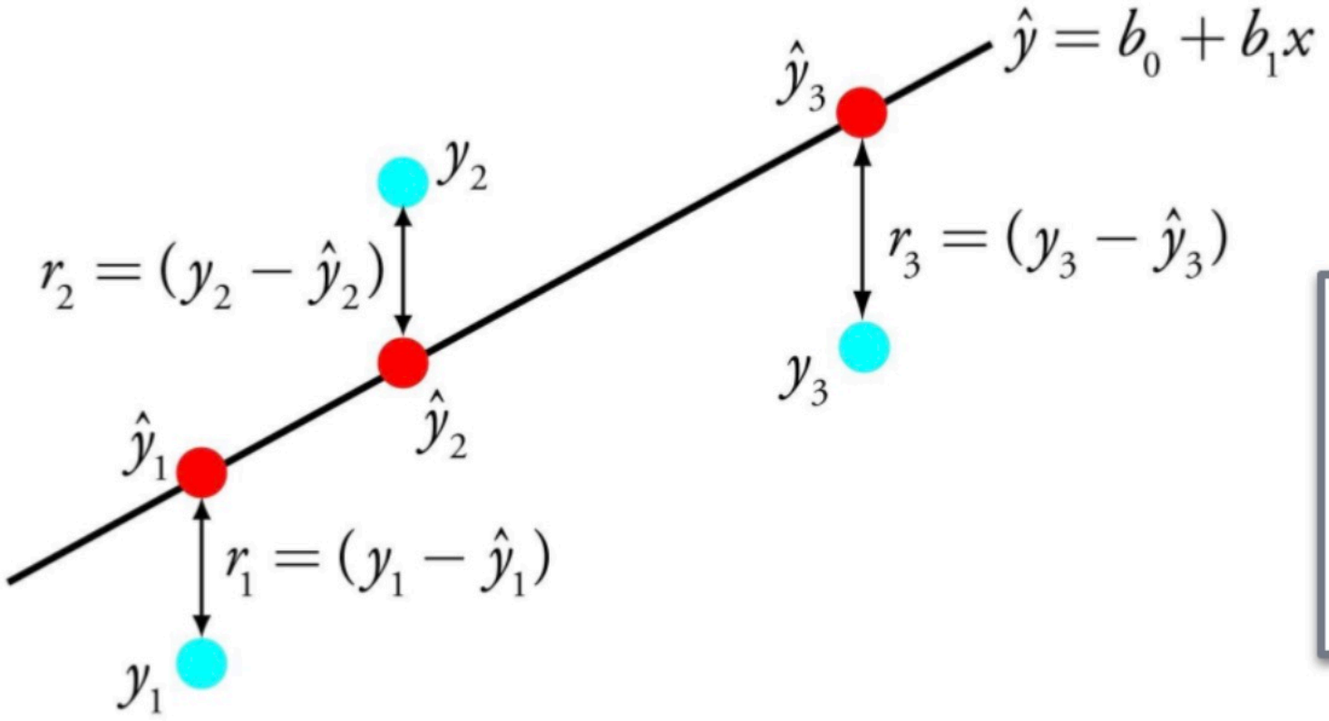


*Mean Absolute Error (MAE)

*Mean Squared Error (MSE)

*Root Mean Square Error (RMSE)

Regression Error Metrics
-Mean Absolute Error (MAE)-

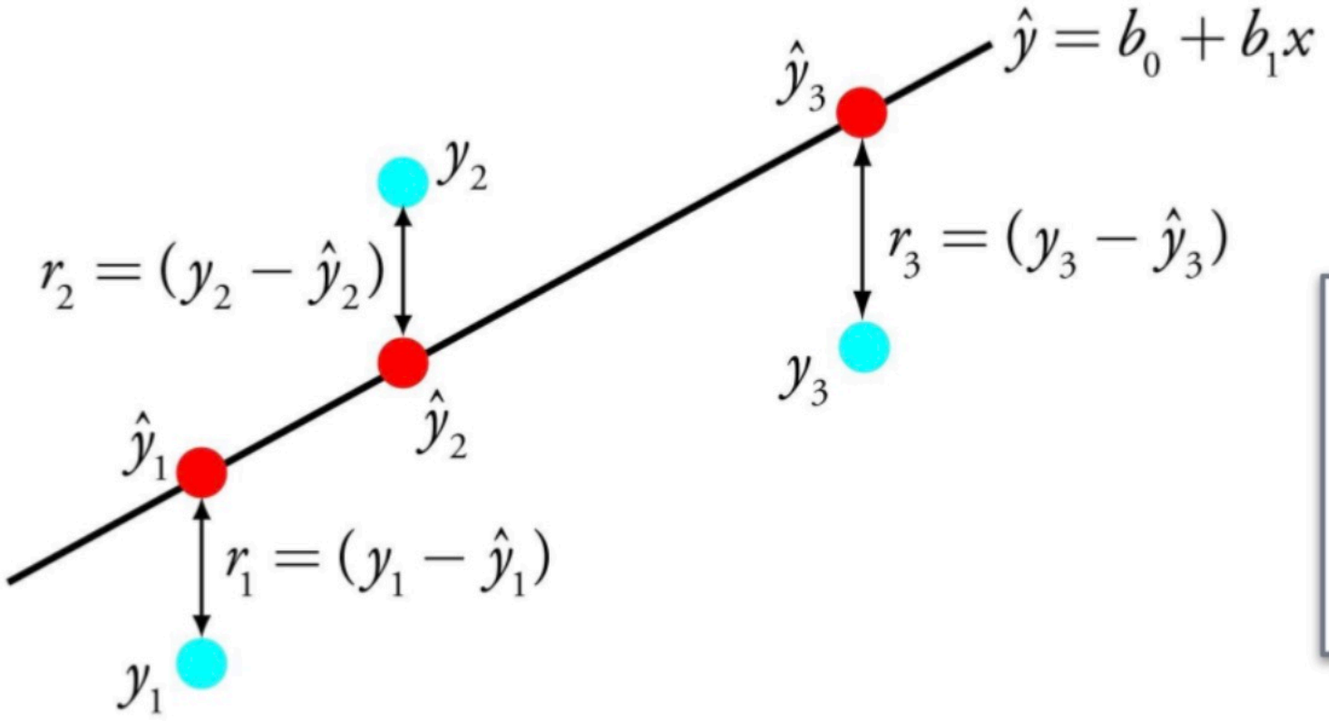


$$\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

* **MAE** is the ***absolute difference*** between the actual target value and the value predicted by the model.

* Not ***penalizes errors.***

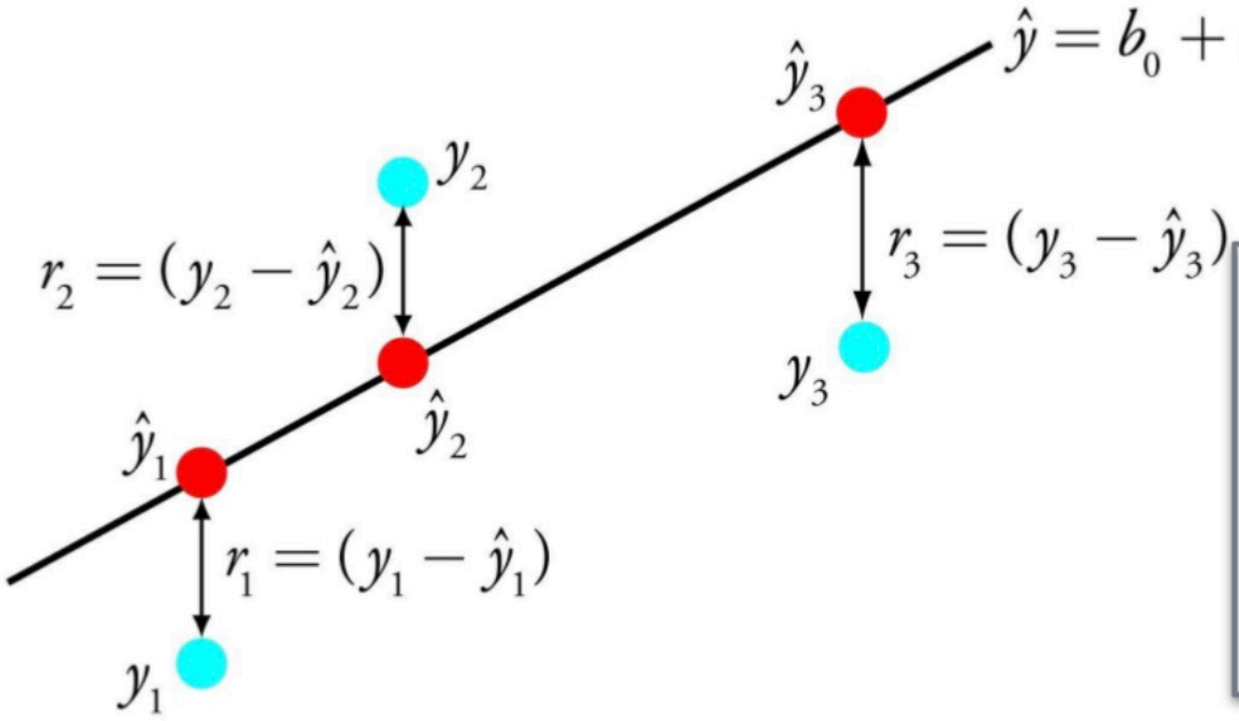
Regression Error Metrics
-Mean Squared Error (MSE)-



$$\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

* As it squares the differences, it ***penalizes*** even a ***small error*** which leads to over-estimation of how bad the model is.

Regression Error Metrics
-Root Mean Squared Error (RMSE)-



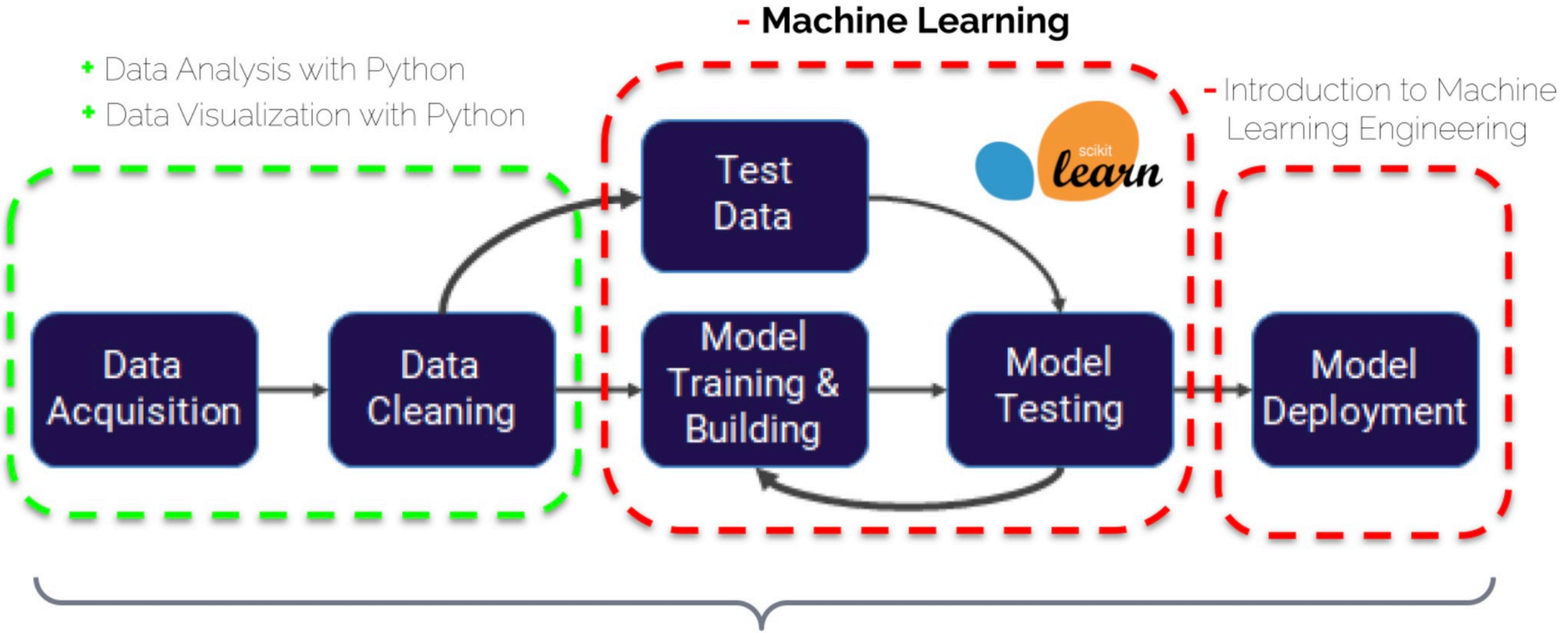
$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

* The errors are first squared before averaging which poses a high penalty on large errors.

* RMSE is useful when ***large errors are undesired.***

Scikit-Learn Library

Machine Learning with Python



Machine Learning Process

Machine Learning with Python

https://scikit-learn.org

`Scikit-learn` is an open source **machine learning library** that supports **supervised** and **unsupervised** learning.

`Scikit-learn` provides various tools for **model fitting, data preprocessing, model selection and evaluation**, and many other utilities.

Machine Learning with Python

conda install scikit-learn

or

pip install scikit-learn

Machine Learning with Python

1. Import

* Evaluation metrics
* Models
* Other necessary functions in sklearn

```
[1]: import numpy as np
     import pandas as pd
     import statsmodels.api as sm
     import statsmodels.formula.api as smf
     import seaborn as sns
     from sklearn.preprocessing import scale
     from sklearn.model_selection import train_test_split, GridSearchCV, cross_val_score
     from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
     from sklearn.metrics import roc_auc_score,roc_curve
     import statsmodels.formula.api as smf
     import matplotlib.pyplot as plt
     from sklearn.neighbors import KNeighborsClassifier
     from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
     from sklearn.linear_model import LogisticRegression
     from sklearn.svm import SVC
     from sklearn.naive_bayes import GaussianNB
     from sklearn import tree
     from sklearn.ensemble import RandomForestClassifier
     from sklearn.ensemble import GradientBoostingClassifier
     from xgboost import XGBClassifier
     from lightgbm import LGBMClassifier
     from catboost import CatBoostClassifier
```

scikit learn

from sklearn.*family* import *model*

Machine Learning with Python

Split the data

Metrics:
Accuracy
Confusion Matrix
Classification Report

Machine Learning Models

```
[1]: import numpy as np
     import pandas as pd
     import statsmodels.api as sm
     import statsmodels.formula.api as smf
     import seaborn as sns
     from sklearn.preprocessing import scale
     from sklearn.model_selection import train_test_split, GridSearchCV, cross_val_score
     from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
     from sklearn.metrics import roc_auc_score,roc_curve
     import statsmodels.formula.api as smf
     import matplotlib.pyplot as plt
     from sklearn.neighbors import KNeighborsClassifier
     from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
     from sklearn.linear_model import LogisticRegression
     from sklearn.svm import SVC
     from sklearn.naive_bayes import GaussianNB
     from sklearn import tree
     from sklearn.ensemble import RandomForestClassifier
     from sklearn.ensemble import GradientBoostingClassifier
```

Machine Learning with Python

2. Splitting Data

```
[34]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=42)
```

3. Model Building and Fitting Data

***model*.fit(X_train, y_train)**

```
[92]: logmodel = LogisticRegression()
      logmodel.fit(X_train,y_train)
```

Machine Learning with Python

4. Predicting Data

predictions = *model*.predict(X_test)

```
[94]: predictions = logmodel.predict(X_test)
```

5. Evaluation of the Model

R2_score = r2_score(actual, pred)

```
R2_score = r2_score(actual, pred)
```

Machine Learning with Python

6. Evaluating Model

cross_validate()

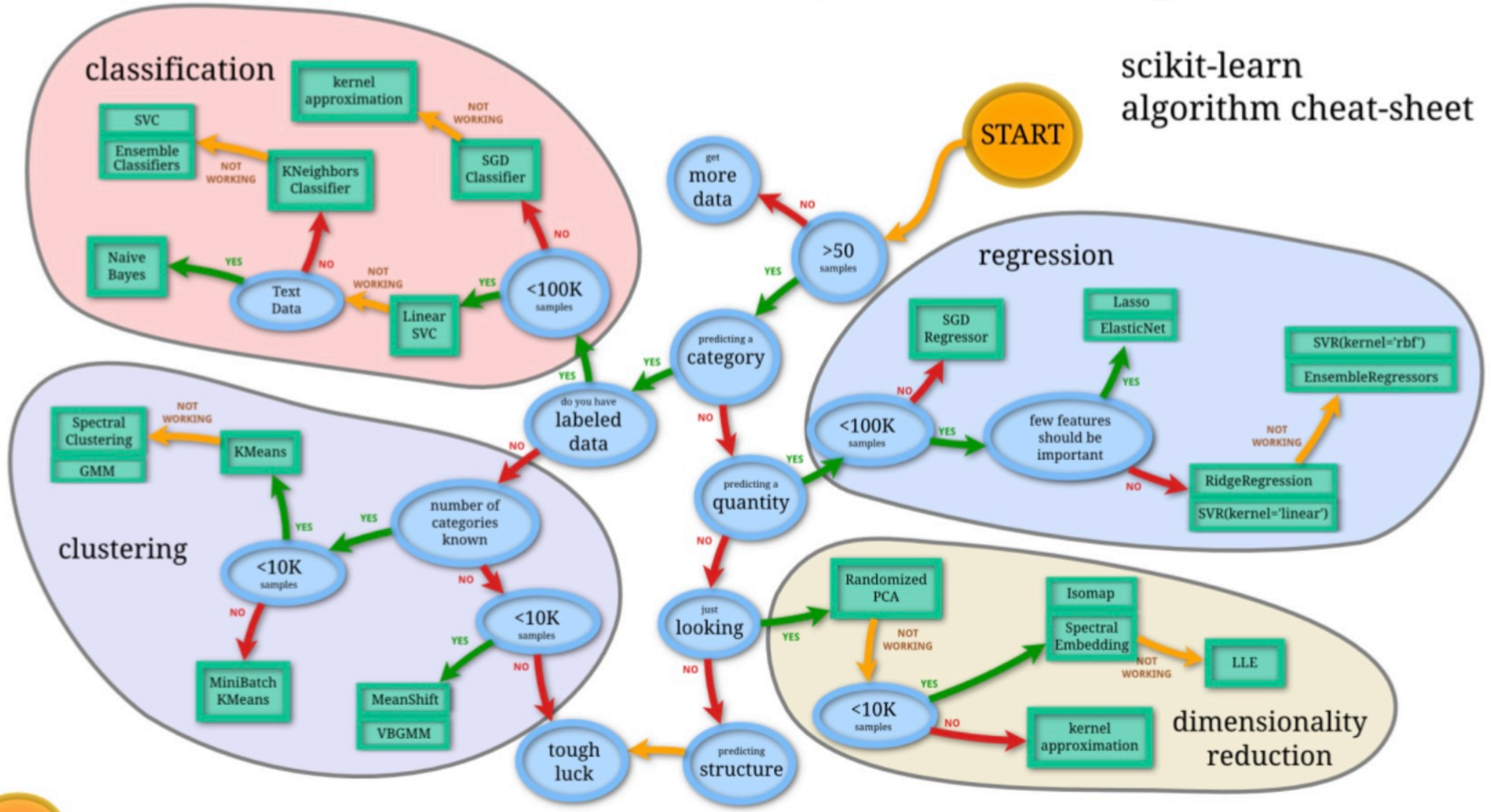
```
In [ ]: scores = cross_validate(model, X_train, y_train, scoring = ['r2', 'neg_mean_absolute_error','neg_mean_squared_error', \
                                                                    'neg_root_mean_squared_error'], cv = 5)
```

7. Hyperparameter Tuning

GridSearchCV()

```
In [214]: grid_model = GridSearchCV(estimator = pipe_elastic, param_grid = param_grid, scoring = 'neg_root_mean_squared_error',
                                    cv =10, verbose =2)
```

Machine Learning with Python



Linear Regression with Python

Be ready for

Linear Regression with Python

Session