Import statements

In [1]:
```
import numpy as np
import pandas as pd
import math as mt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

Function

In [2]:
```
def generate_n_samples(n):
    np.random.seed(1234)
    spvar_x = np.random.uniform(-3,3,n)
    eps = np.random.uniform(0,1,n)
    res_y = 8 * np.sin(spvar_x) + eps
    df = pd.DataFrame({'X':spvar_x,'Y':res_y})
    #print(df)
    return df
```

In [3]:
```
def model_training(train_s,test_s,deg):
    training_sample = generate_n_samples(train_s)
    #print(training_sample)
    x_train = training_sample[['X']]
    y_train = training_sample[['Y']]

    testing_sample = generate_n_samples(test_s)
    x_test = testing_sample[['X']]
    y_test = testing_sample[['Y']]

    poly_feat = PolynomialFeatures(degree=deg)
    X_train_pf = poly_feat.fit_transform(x_train)
    X_test_pf = poly_feat.fit_transform(x_test)

    model = LinearRegression()
    model.fit(X_train_pf,y_train)

    y_pred = model.predict(X_test_pf)
    MSE = mean_squared_error(y_test,y_pred)

    return MSE
```

In [ ]:
```

```

Calling the functions

In [4]:
```
mse_degree_3_tr50_te_10000 = model_training(50, 10000, 3)
mse_degree_15_tr50_te_10000 = model_training(50, 10000, 15)
```

In [5]:
```
mse_degree_3_tr10000_te_10000 = model_training(10000, 10000, 3)
mse_degree_15_tr10000_te_10000 = model_training(10000, 10000, 15)
```

In [6]:
```
print("MSE for Degree 3 with training set of size 50 and testing set of size 10000:
print("MSE for Degree 15 with training set of size 50 and testing set of size 10000
print("MSE for Degree 3 with training set of size 10000 and testing set of size 100
print("MSE for Degree 15 with training set of size 10000 and testing set of size 10
```

```
MSE for Degree 3 with training set of size 50 and testing set of size 10000: 0.31578
39420145649
MSE for Degree 15 with training set of size 50 and testing set of size 10000: 3.2623
437804238873
MSE for Degree 3 with training set of size 10000 and testing set of size 10000: 0.27
073591784351525
MSE for Degree 15 with training set of size 10000 and testing set of size 10000: 0.0
8413596045618944
```

In [ ]:

Best prediction rule

In our case we know the relation between the predictor variable and outcome variable Hence in our case the best possible prediction rule will be f(x)=8sin(x)

Obtaining test MSE for the best prediction rule

In [7]:
```
test_data_for_best_MSE = generate_n_samples(10000)
X_test = test_data_for_best_MSE[['X']]
y_true = test_data_for_best_MSE[['Y']]
```

In [8]:
```
y_pred_best_rule = 8 * np.sin(X_test['X'])
```

In [9]:
```
mse_best_rule = mean_squared_error(y_true, y_pred_best_rule)

print("MSE for the best prediction rule:", mse_best_rule)
```

```
MSE for the best prediction rule: 0.3348642696474157
```

In [ ]:

BIAS and VARIANCE of the results

In [10]:
```
print("MSE for Degree 3 with training set of size 50 and testing set of size 10000:
print("MSE for Degree 15 with training set of size 50 and testing set of size 10000
print("MSE for Degree 3 with training set of size 10000 and testing set of size 100
print("MSE for Degree 15 with training set of size 10000 and testing set of size 10
```

```
MSE for Degree 3 with training set of size 50 and testing set of size 10000: 0.31578
39420145649
MSE for Degree 15 with training set of size 50 and testing set of size 10000: 3.2623
437804238873
MSE for Degree 3 with training set of size 10000 and testing set of size 10000: 0.27
073591784351525
MSE for Degree 15 with training set of size 10000 and testing set of size 10000: 0.0
8413596045618944
```

In [ ]:

For small training set:- the models with degree 15 are likely to have higher variance due to overfitting, which can result in a higher MSE compared to degree 3 models. The small training set size limits the ability of complex models to generalize well. The high-degree models may fit the training data well (low bias) but could suffer from high variance, leading to a higher MSE on the test set. For large training set:- the degree 3 model might have higher bias, as it may not capture the underlying complexity well. The degree 15 model could have lower bias but potentially higher variance, leading to a trade-off. With a larger training set, the degree 15 model might be able to leverage the additional data to reduce variance and provide better predictions, resulting in a lower MSE compared to the degree 3 model.

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: