REPORT

Team 93
2019101102
2019101104

Task-1:

Linear regression.fit():

LinearRegression creates the object that represents the model, while .fit() trains, or fits, the model and returns it. With linear regression, fitting the model means determining the best intercept (model.intercept_) and slope (model.coef_) values of the regression line.

```
from sklearn.linear_model import LinearRegression
regression = LinearRegression()
regression.fit(X_train, y_train)
```

Linear regression can be considered a Machine Learning algorithm that allows us to map numeric inputs to numeric outputs, fitting a line into the data points.

In Linear regression implementation, we need to import the LinearRegression class, instantiate it, and call the fit() method along with our training data. This is about as simple as it gets when using a machine learning library to train on your data. The linear regression model basically finds the best value for the intercept and slope, which results in a line that best fits the data.

With .fit(), we can calculate the optimal values of the weights weights and intercept, using the existing input and output (x and y) as the arguments. In other words, .fit() fits the model. It returns self, which is the variable model itself. So, we can write it as:

```
model = LinearRegression().fit(x, y)
```

Instead of

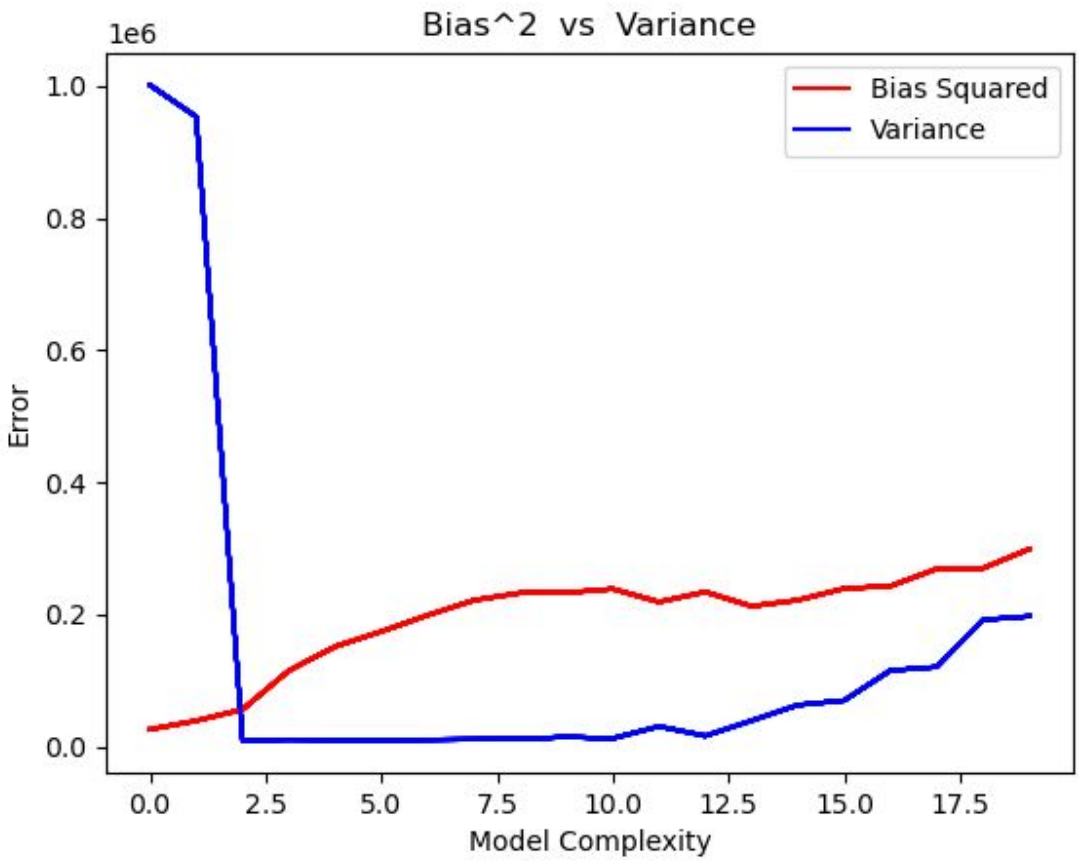
```
model.fit(x, y)
```

Task-2:

CALCULATING BIAS AND VARIANCE:

Degree	Bias	variance
1	1000.8408967707331	25999.09300998779
2	976.6453442392459	39105.83381326901
3	97.63884506839366	56095.89320974702
4	102.89945790450103	114907.29153007835
5	99.76194246799936	151434.02779626832
6	102.10751883564524	174226.74553968332
7	99.76194246799936	198849.51451574033
8	104.86366236363914	221553.13460753072
9	107.51504717137716	232604.94916099348
10	120.31953515217963	233053.77824789443
11	111.42912440429865	238706.03580934933
12	174.78917424116088	219014.0912557728
13	127.17664974755856	234166.2491915726
14	198.0684099332389	212545.2558827084
15	250.85526665945815	221715.00801977847
16	264.2375916648397	239357.86903651836
17	339.46932380003784	242994.25846773217
18	347.24687300366946	269049.8634861061
19	438.1546508383121	270101.7446019015
20	444.16778013214315	299027.3120568435

Graph:



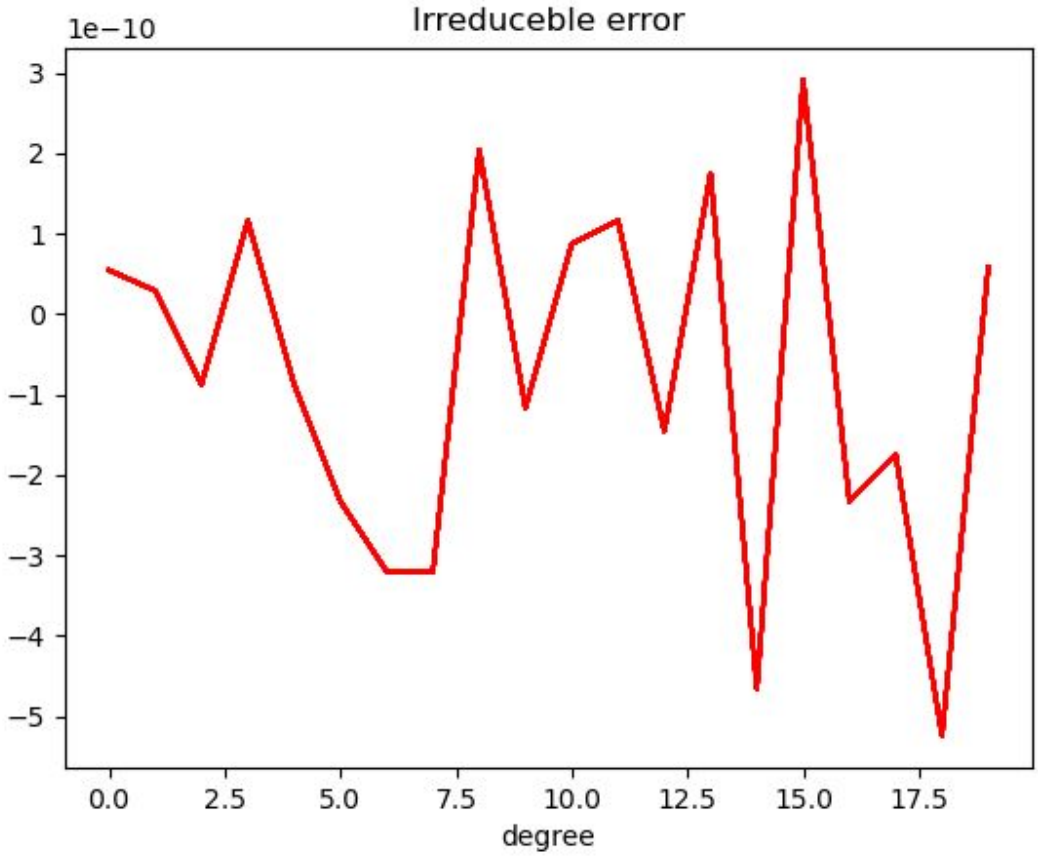
We observe that, if complexity of the model increases then bias decreases and variance increases.

As the degree of polynomial increases, the model extracts more information as there is an increase in the number of features(variables). Thus results in low bias and high variance with increase in model complexity. But we cannot generalize the model because we do not know the test data and the model performance can be poor.

Similarly, for lower degree polynomials, the model shows high bias and low variance and performance can be poor because the lower degree polynomial cannot access all the features or variables of training data.

Task-3:

Irreducible Error:



Degree	Error
1	5.4569682106375694e-11
2	2.9103830456733704e-11
3	-8.731149137020111e-11
4	1.1641532182693481e-10

5	-8.731149137020111e-11
6	-2.3283064365386963e-10
7	-3.2014213502407074e-10
8	-3.2014213502407074e-10
9	2.0372681319713593e-10
10	-1.1641532182693481e-10
11	8.731149137020111e-11
12	1.1641532182693481e-10
13	1.4551915228366852e-10
14	1.7462298274040222e-10
15	-4.656612873077393e-10
16	2.9103830456733704e-10
17	-2.3283064365386963e-10
18	-1.7462298274040222e-10
19	-5.238689482212067e-10
20	5.820766091346741e-11

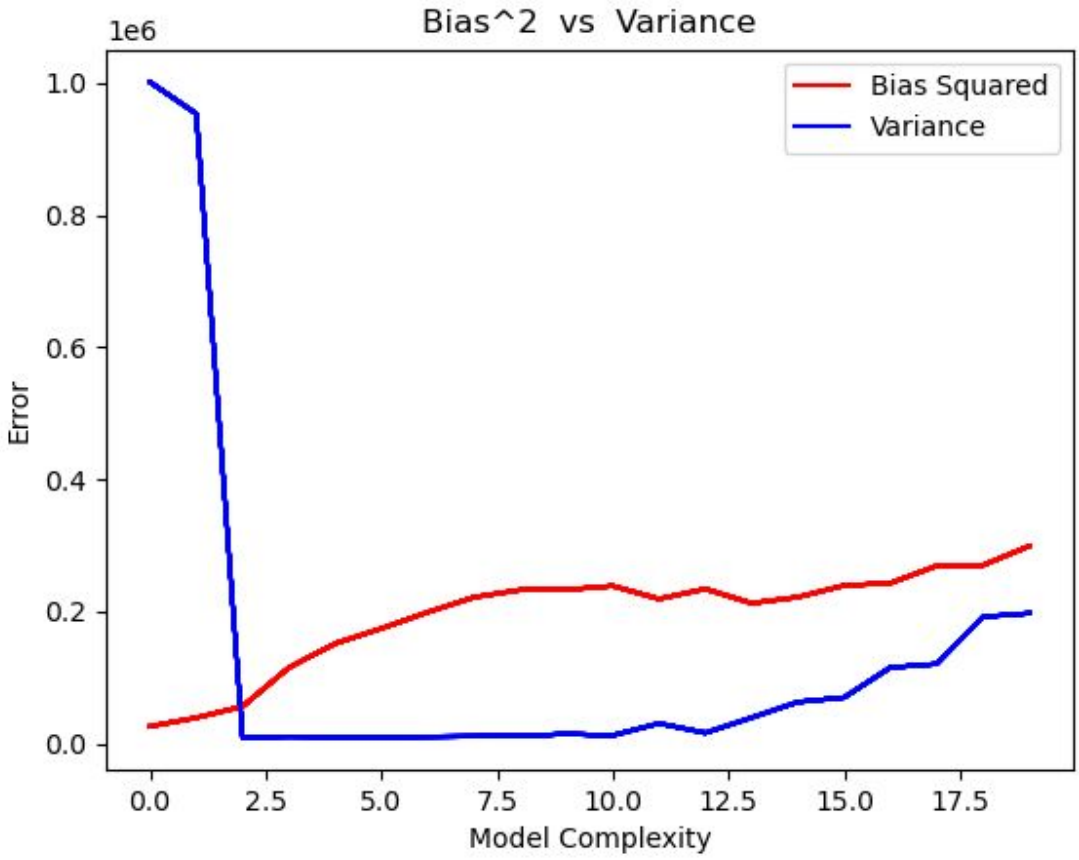
Irreducible Error: Errors which can't be removed no matter what algorithm you apply. These errors are caused by unknown variables that are affecting the independent/output variable but are not one of the dependent/input variables while designing the model.

$$E[(f(x) - \hat{f}(x))^2] = \text{Bias}^2 + \sigma^2 + \text{Variance}$$

$$\sigma^2 = E[(f(x) - \hat{f}(x))^2] - (\text{Bias}^2 + \text{Variance})$$

Task-4:

Plotting Bias^2 and variance



Overfitting occurs when your model learns too much from training data and isn't able to generalize the underlying information. When this happens, the model is able to describe training data very accurately but loses precision on every dataset it has not been trained on.

If the number of variables/features in a model increases then the model extracts more information which might be suitable to work well with training data because of generalizing and an increase in identifying patterns but not necessarily with the test data as the data sets are unknown. This results in low bias and high variance with the increase in complexity (here degree of polynomial) and the model can perform poor and thus overfitting

Underfitting occurs when a model is too simple . informed by too few features or regularized too much which makes it inflexible in learning from the dataset. Simple learners tend to have less variance in their predictions but more bias towards wrong outcomes.

If the number of variables/features in a model decreases then the model could not extract required information that needs to be worked on training data. This results in high bias and low variance with the decrease in complexity(here degree of polynomial) and the model can perform poorly and thus underfitting.