Machine Learning HW1

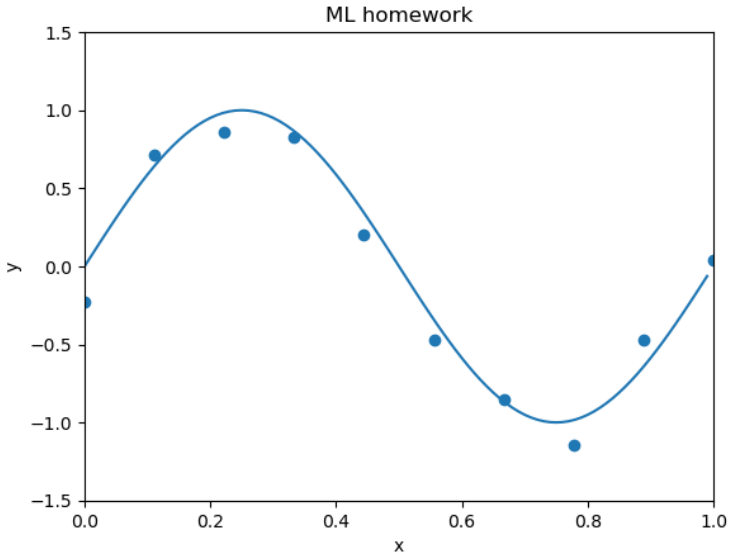
202211167 장지원

1. Plot 10 samples, spaced uniformly in range [0, 1], with the function sin(2πx) with a Gaussian noise like below.

   1번 문제에서는 sin(2πx)에 Gaussian noise를 추가하여 10개의 sample을 찍어야 한다. 코드는 다음과 같이 작성하였다. 코드를 실행하면 [그래프1]과 같은 그래프가 나오게 된다.

```
# 그래프 생성
x = np.arange(0, 1, 0.01)
y = np.sin(2*np.pi*x)
plt.plot(x,y)
plt.xlim([0, 1])
plt.ylim([-1.5,1.5])
plt.xlabel("x")
plt.ylabel("y")
plt.title("ML homework")

# sample 생성
num_samples = 10
x_values = np.linspace(0, 1, num_samples)  # 샘플 추출
gaussian = np.random.normal(0, 0.1, num_samples)  # 노이즈 생성
y_values = np.sin(2 * np.pi * x_values) + gaussian # 노이즈를 더해주기
plt.scatter(x_values, y_values)
```



[그래프1] – sin graph with 10 samples

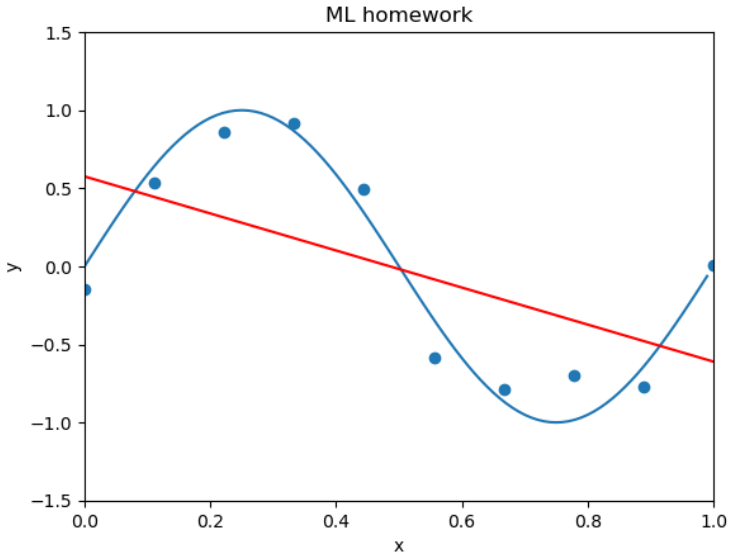
2. Generate regression lines with polynomial basis function with order 1, 3, 5, 9, and 15.

a. order 1

   sklearn라이브러리의 *LinearRegression()*함수를 이용해서 모델을 만들어보았다. 이후 Linear regression이 요구하는 형태로 입력 데이터를 변환해주고, 이를 바탕으로 fitting 시켜보았다. 결과는 [그래프2]와 같다.

```
# 1-a. linear regression
model = LinearRegression()
model.fit(X=x_values.reshape(-1, 1), y=y_values)
data = x_values.reshape(-1, 1)
```

```
pred = model.predict(data)
plt.plot(x_values, pred, color='red')
```



[그래프2] - regression lines with polynomial basis function with order 1

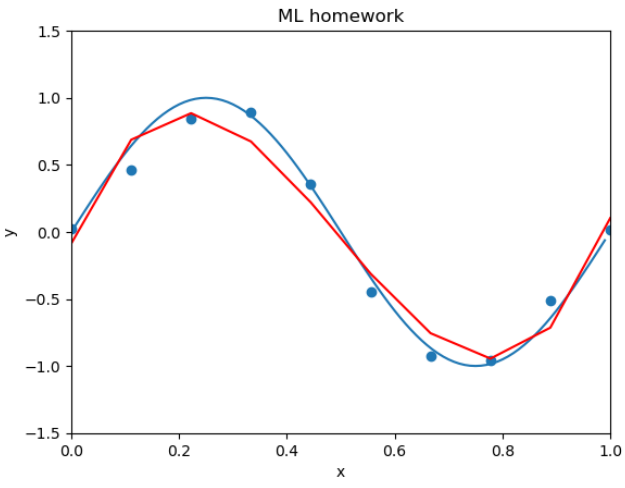
b. order 3, 5, 9, 15

poly_features = PolynomialFeatures(degree=3, include_bias=False)

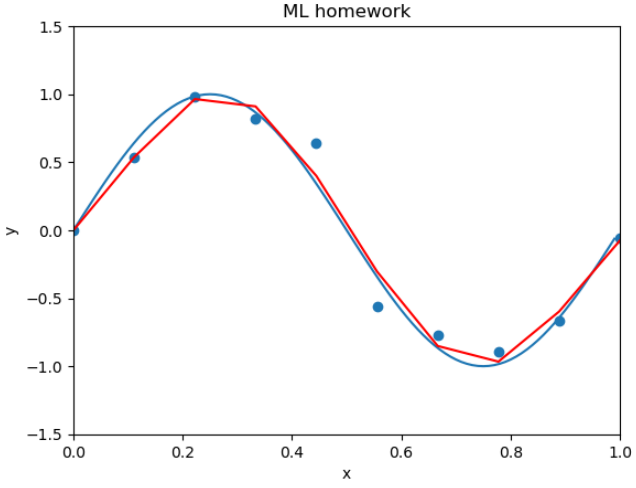
x_poly = poly_features.fit_transform(x_linear)

두 라인을 활용해서 입력 데이터를 Polynomial regression이 요구하는 형태로 바꾸어준다. 다항식의 차수는 *PolynomialFeatures(degree=3, include_bias=False)*의 degree 파라미터를 수정하면 된다. 결과는 [그래프3]~[그래프6]과 같다.

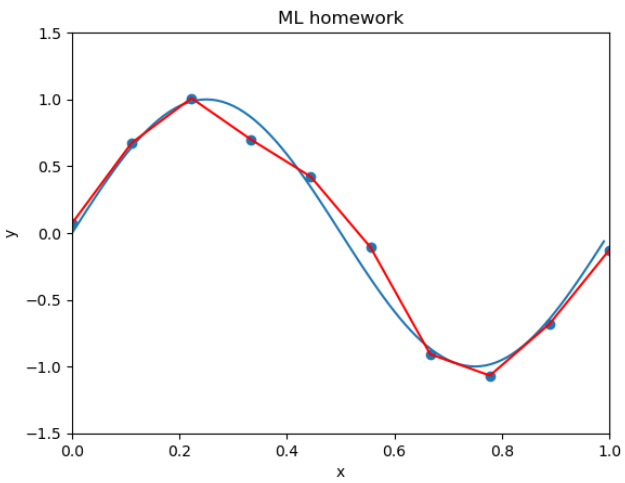
```
# 1-b. polynomial regrssion (3)
model = LinearRegression()
poly_features = PolynomialFeatures(degree=3, include_bias=False) #
PolynomialFeatures(차수, 편향 변수(=1) 확인 후 추가)
x_linear = x_values.reshape(-1, 1) # 2 차원 배열로의 변환
x_poly = poly_features.fit_transform(x_linear) # 다항 특성 가지게 변환
model.fit(x_poly, y_values)
pred = model.predict(x_poly) # 예측값 계산
plt.plot(x_values, pred, color='red') # 결과 시각화
```



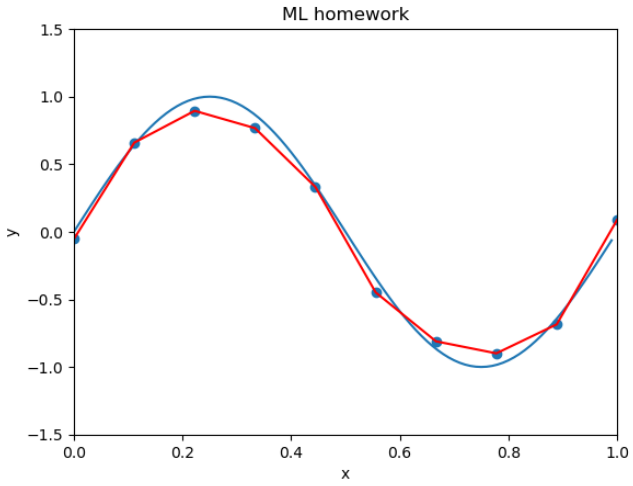
[그래프3] - regression lines with polynomial basis function with order 3



[그래프4] - regression lines with polynomial basis function with order 5



[그래프5] - regression lines with polynomial basis function with order 9



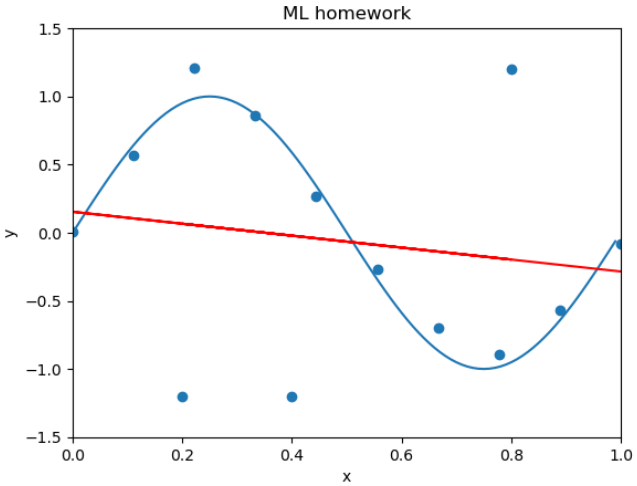
[그래프6] - regression lines with polynomial basis function with order 15

차수가 높아질수록 입력 데이터에 더 많이 fitting되는 것을 확인할 수 있다. test data를 통해 결과를 확인해보아야 하겠지만 order가 9이상인 경우에는 model이 overfitting된 모습을 보일 것이다.

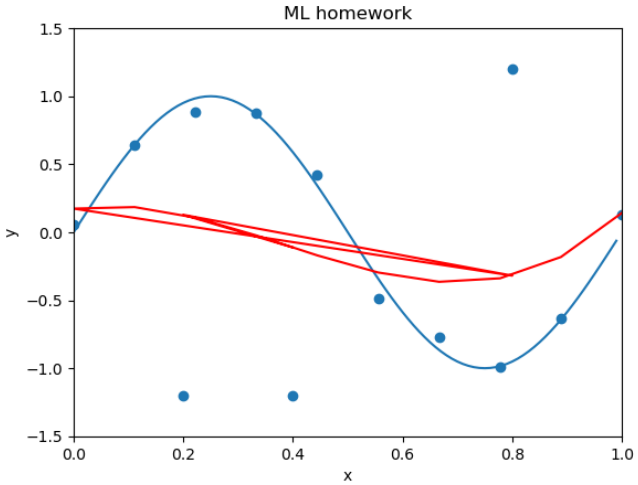
3. Add 2 or 3 points of exceptional outliers that do not follow sin(2πx) and then generate regression lines with polynomial basis function with order 1, 3, 5, 9, and 15.

sin(2πx)그래프를 따르지 않는 10개의 outlier를 추가해보았다. 이후 이 데이터들을 바탕으로 다시 polynomial regression을 진행해 보았다. 결과는 [그래프7]~[그래프11]과 같았다.

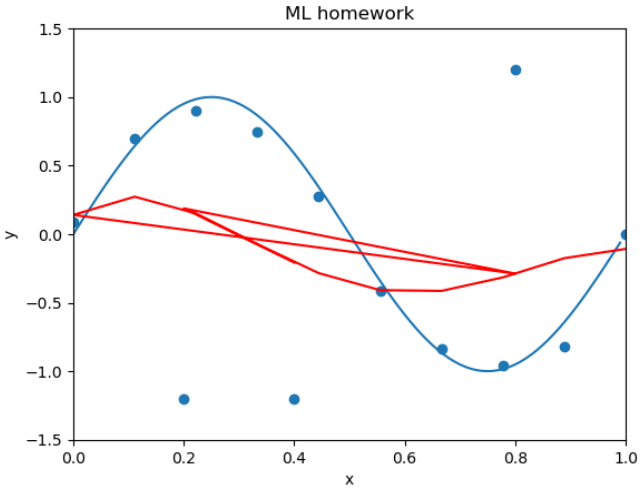
```
x_values=np.append(0.8,x_values)
x_values=np.append(0.2,x_values)
x_values=np.append(0.4,x_values)
y_values=np.append(1.2,y_values)
y_values=np.append(-1.2,y_values)
y_values=np.append(-1.2,y_values)
```



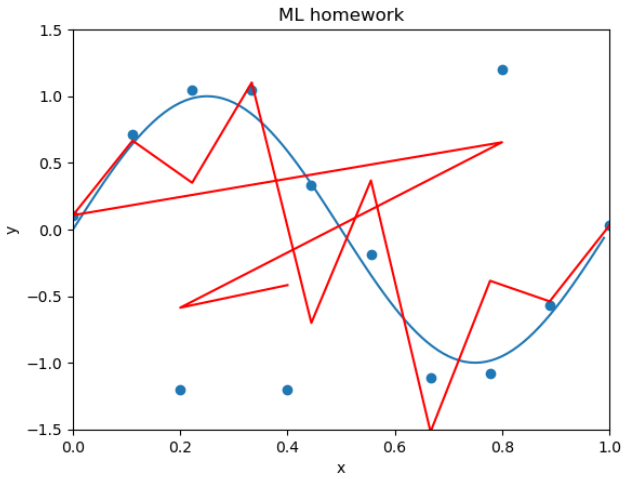
[그래프7] - regression lines with polynomial basis function with order 1



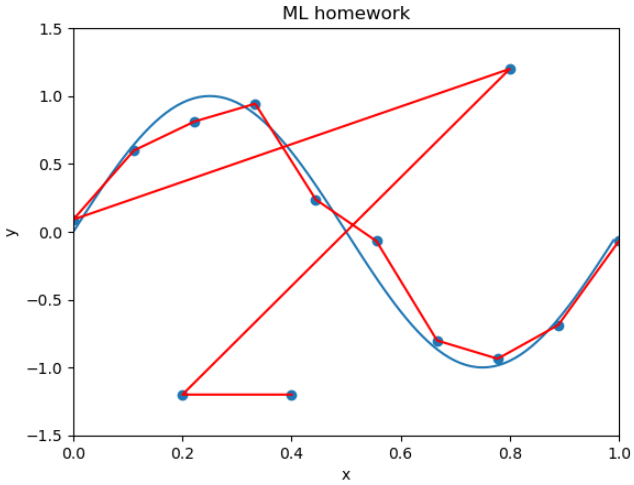
[그래프8] - regression lines with polynomial basis function with order 3



[그래프9] - regression lines with polynomial basis function with order 5



[그래프10] - regression lines with polynomial basis function with order 9



[그래프11] - regression lines with polynomial basis function with order 15

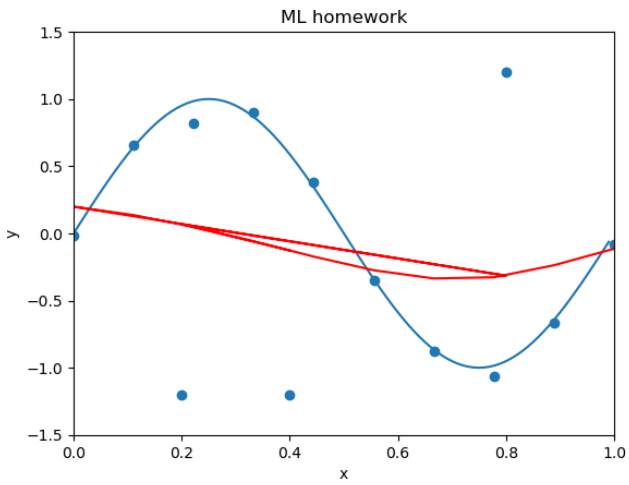
Outlier가 있을 때는 더 심하게 fitting되는 것을 확인할 수 있었다.

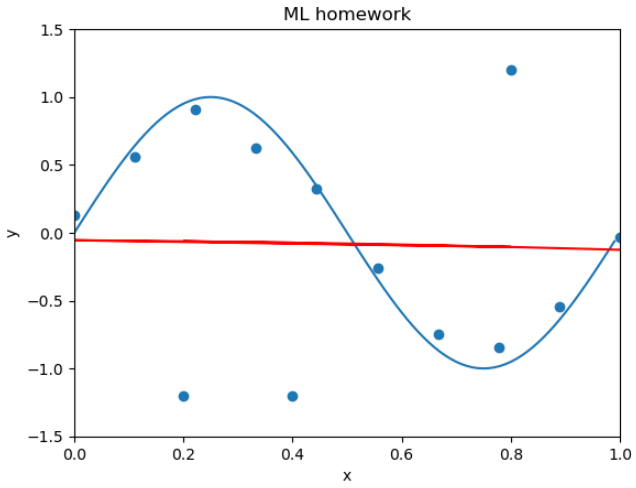
4. For the case including the outliers, generate the regression lines with the L2 regularization term with order 9 and 15. Show how the lines are changed with respect to λ . Generate the regression lines with the L1 regularization term and compare the lines with L2 regularization.

L2 regularization은 sklearn라이브러리에서 Ridge() 함수로써 제공하고 있다. 파라미터인 λ이 커지면 모델이 단순해지고, λ이 작아지면 모델이 복잡해진다. 이는 [그래프12]~[그래프15]를 통해 확인할 수 있다. 우리는 이 방식을 사용함으로써 위에서의 overfitting 문제를 해결할 수 있다.

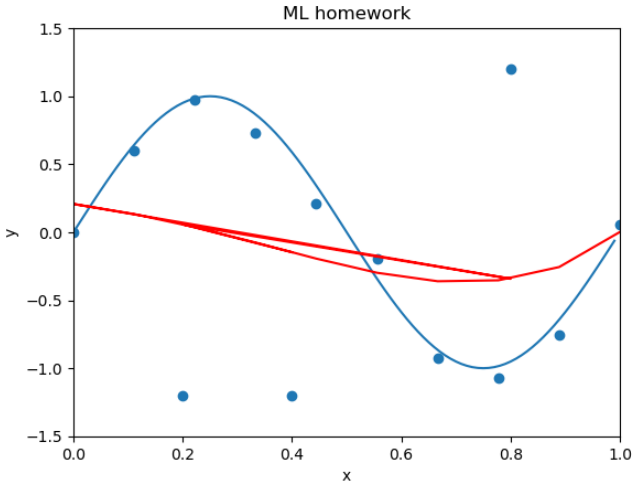
```
x_values=np.append(0.8,x_values)
x_values=np.append(0.2,x_values)
x_values=np.append(0.4,x_values)
y_values=np.append(1.2,y_values)
y_values=np.append(-1.2,y_values)
y_values=np.append(-1.2,y_values)
plt.scatter(x_values, y_values)
model = Ridge(0.01)
poly_features = PolynomialFeatures(degree=9, include_bias=False)
#PolynomialFeatures(차수, 편향 변수(=1) 확인 후 추가)
x_linear = x_values.reshape(-1, 1) # 2 차원 배열로의 변환
x_poly = poly_features.fit_transform(x_linear) # 다항 특성 가지게 변환
```

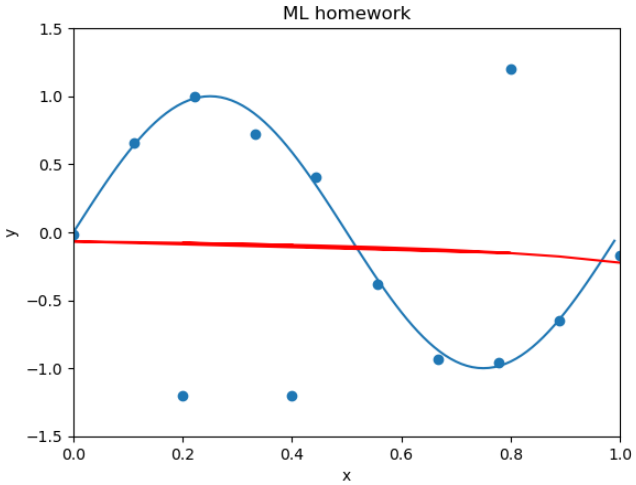
```
model.fit(x_poly, y_values)
pred = model.predict(x_poly) # 예측값 계산
plt.plot(x_values, pred, color='red') # 결과 시각화
```





[그래프12] - regression lines with the L2 regularization term with order 9(λ=0.01) [그래프13] - regression lines with the L2 regularization term with order 9(λ=10)



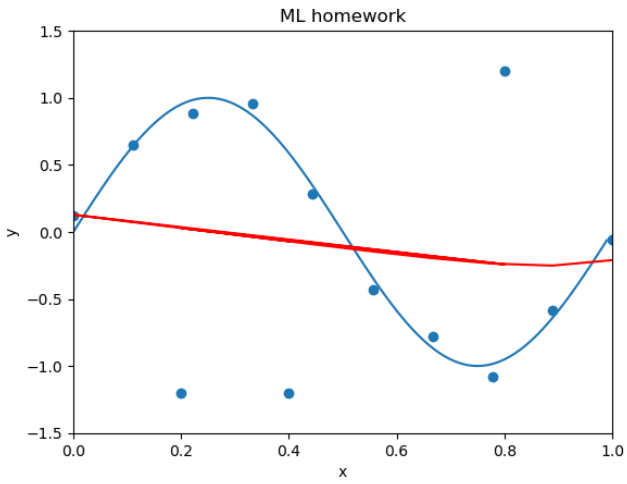


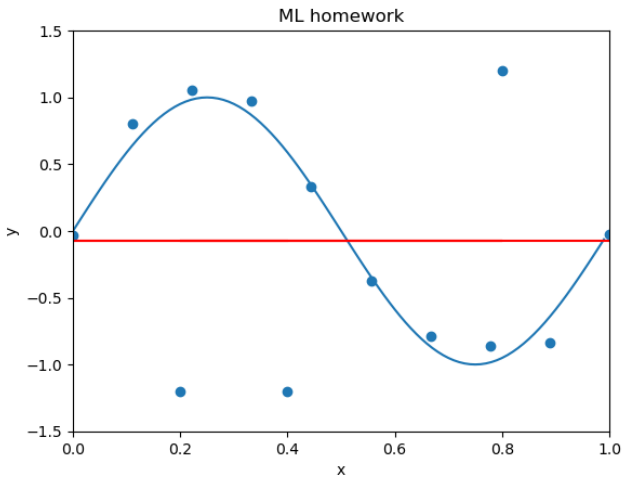
[그래프14] - regression lines with the L2 regularization term with order 15(λ=0.01) [그래프15] - regression lines with the L2 regularization term with order 15(λ=10)

실제로 λ이 작아질수록 모델이 복잡해지는 것을 확인할 수 있다. 또한 2번과 3번 결과에 비해 overfit 문제가 많이 해결된 것을 확인할 수 있다.

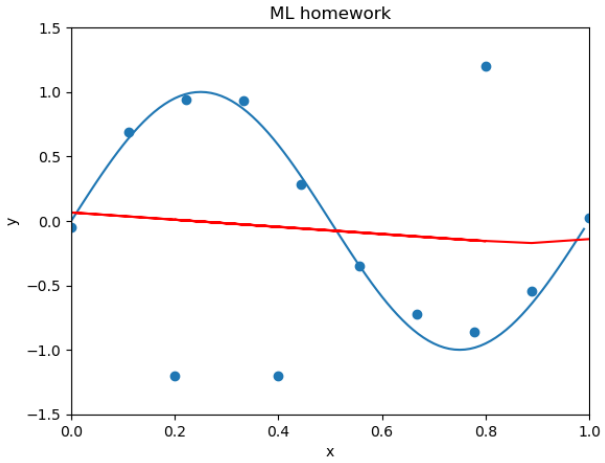
L1 regularization은 sklearn라이브러리에서 Lasso() 함수로써 제공하고 있다. L1 regualrization의 결과는 [그래프16]~[그래프19]를 통해서 확인할 수 있다. 우리는 아까와 마찬가지로 이 방식을 사용함으로써 위에서의 overfitting 문제를 해결할 수 있다.

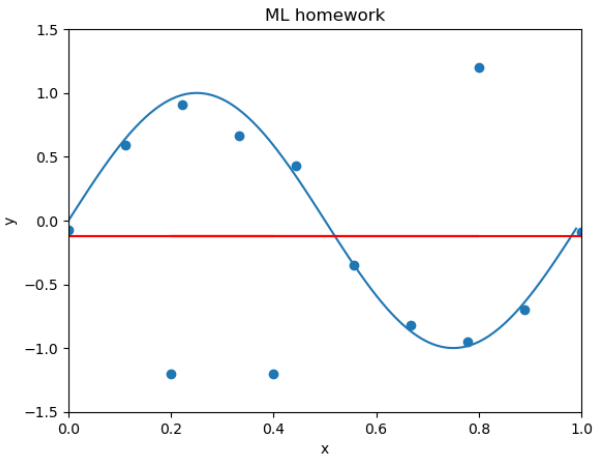
```
model = Lasso(0.01)
```





[그래프16] - regression lines with the L1 regularization term with order 9(λ=0.01) [그래프17] - regression lines with the L1 regularization term with order 9(λ=10)





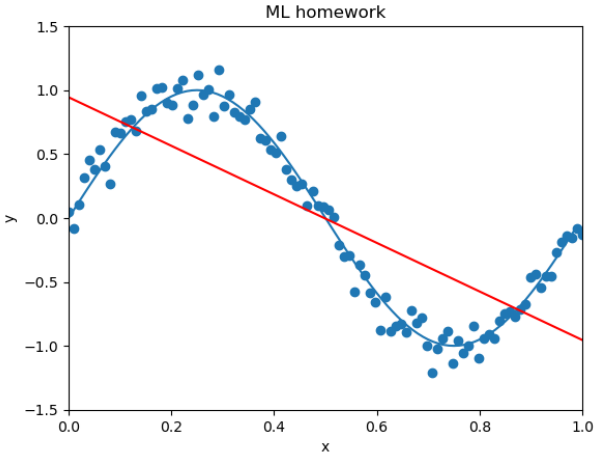
[그래프18] - regression lines with the L1 regularization term with order 15(λ=0.01) [그래프19] - regression lines with the L1 regularization term with order 15(λ=10)

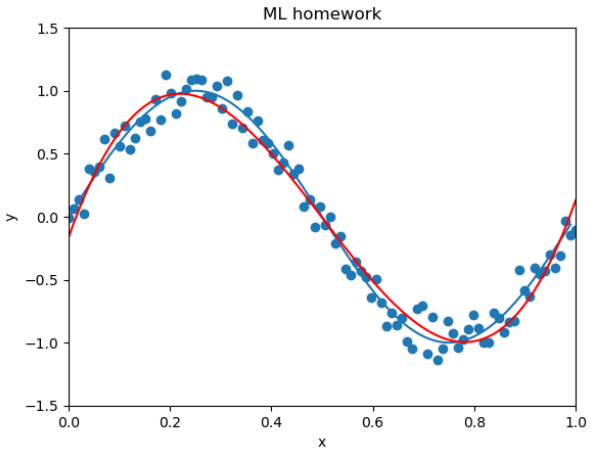
L1 regularization에서도 overfit 문제가 해결됨을 확인할 수 있다. 또한 우리는 L1 regualrization이 L2 regualarization 보다 조금 더 단순한 형태임도 확인할 수 있다. 어떤 것이 정답인지는 모른다. 때에 따라 맞는 방식을 사용하는 것이 좋다.

5. Plot 100 samples with the function sin(2πx) instead of 10 samples, and then generate the regression lines with order 1, 3, 5, 9, and 15.

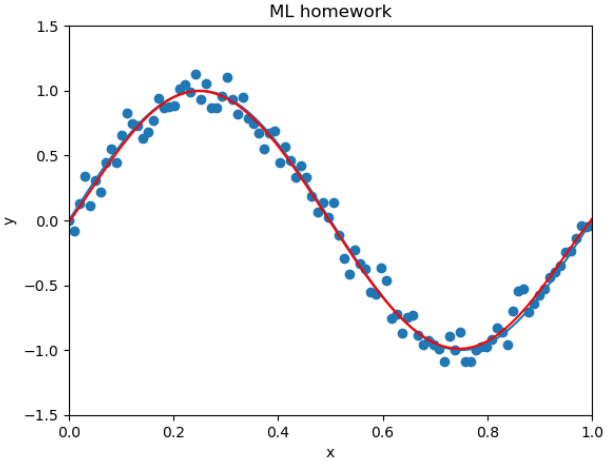
sample을 100개로 늘려보았다. 데이터가 많아짐에 따라 overfit되는 문제가 해결될 것임을 예상해볼 수 있다. 이는 [그래프20]~[그래프24]를 통해서 알아볼 수 있다.

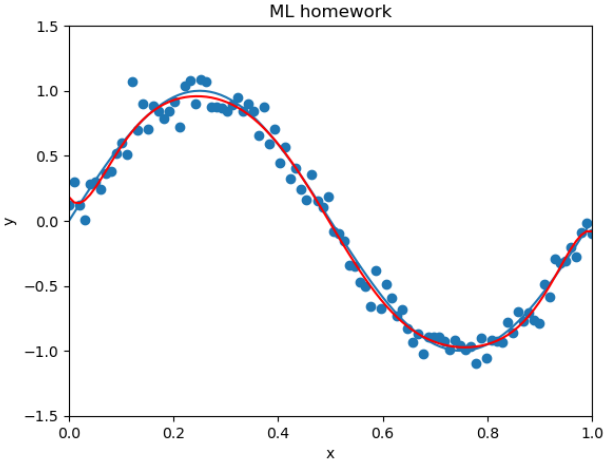
```
num_samples = 100
```



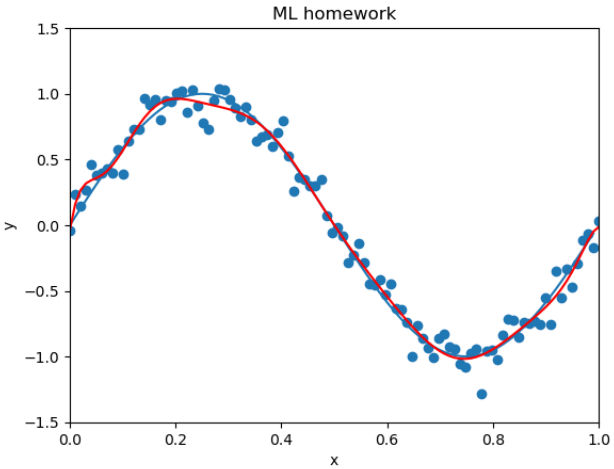


[그래프20] - regression lines with polynomial basis function with order 1 [그래프21] - regression lines with polynomial basis function with order 3





[그래프22] - regression lines with polynomial basis function with order 5 [그래프23] - regression lines with polynomial basis function with order 9



[그래프24] - regression lines with polynomial basis function with order 15

2,3번에서의 Overfitting 문제가 해결되었음을 확인할 수 있다.