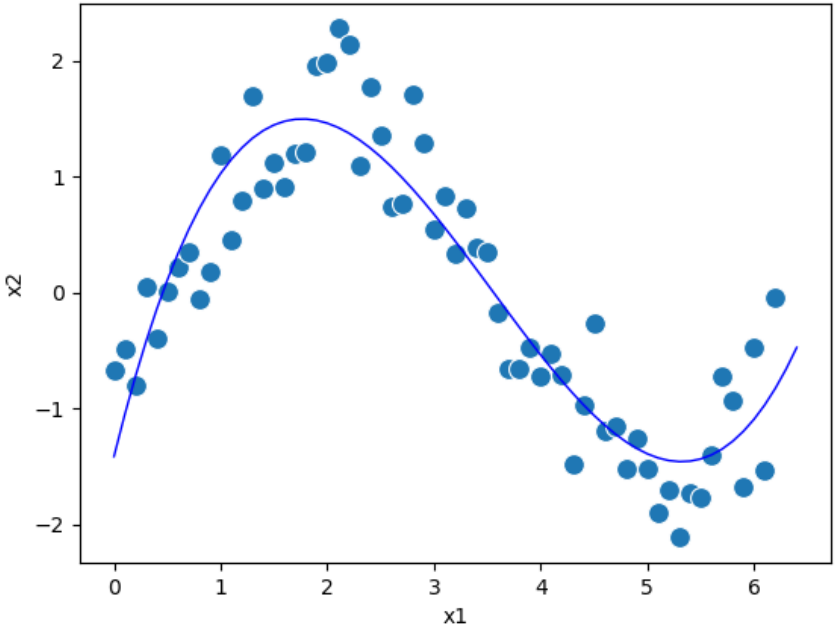
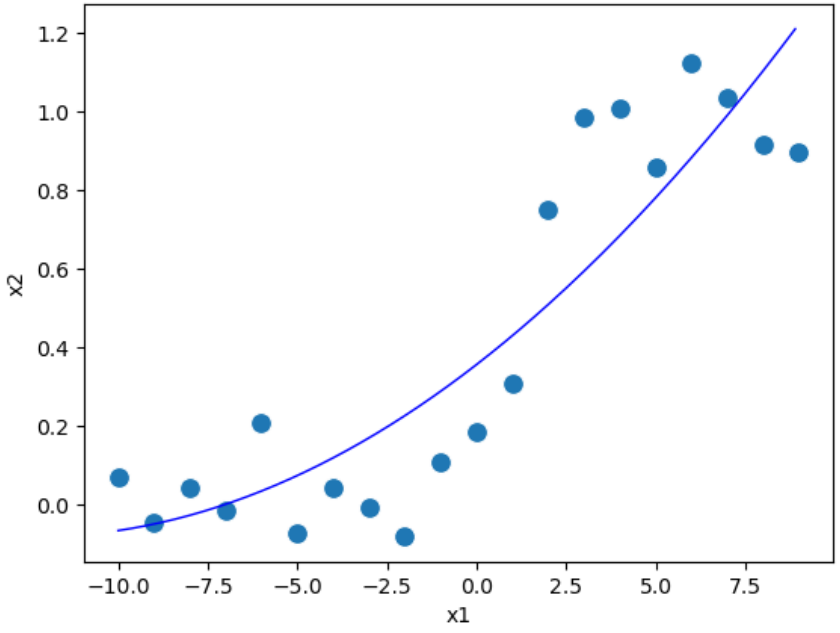
Regression 1 : The data is noisy bi - quadratic polynomial function (degree - 4).




```
x = np.arange(0.0, 6.5, 0.1)
sin = poly_quadratic_function(x, *s.x)
plt.plot(x, sin, color='blue', linewidth=1)
sns.scatterplot(data=r1, x='x1', y='x2', s=100)
plt.show()
```

- To analyze the above noisy data, I created a bi-quadratic polynomial function of degree 4.
- I computed the values and slope to match the appropriate function and minimized the distance between the provided data points.

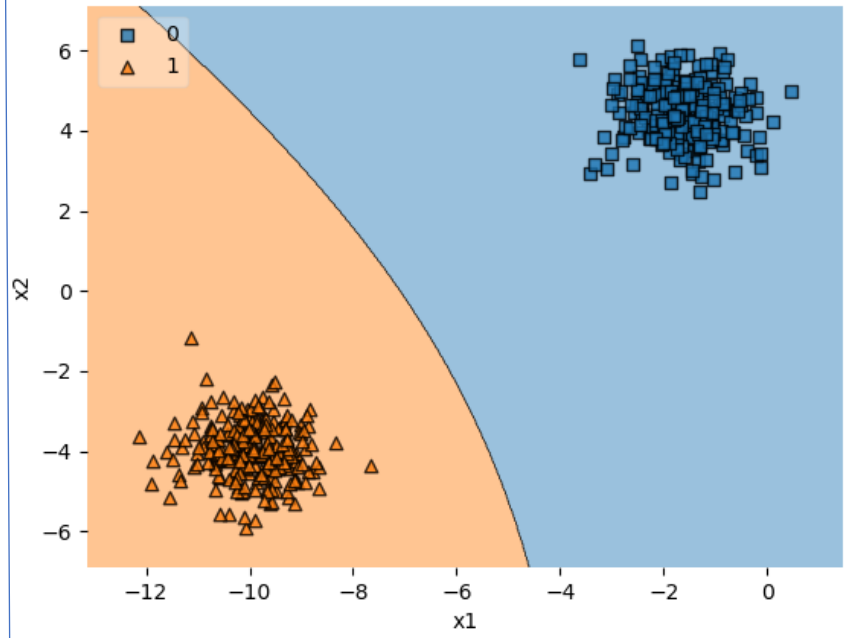
Regression 2 : The data is non-noisy quadratic polynomial function with degree 3




```
x = np.arange(-10.0, 9.0, 0.1)
parabola_r2 = quadratic_function(x, *m.x)
plt.plot(x, parabola_r2, color='blue', linewidth=1)
sns.scatterplot(data=r2, x='x1', y='x2', s=100)
plt.show()
```

- In order to achieve a good fit, I applied a quadratic function since the datapoints were steadily increasing.
- I calculated the values and slope of the optimal fit and minimized the distance between the datapoints.

Classification : The data is non noisy classified data.




```
xc, yc = np.c_[c['x1'], c['x2']], c.label
from sklearn.svm import SVC
from mlxtend.plotting import plot_decision_regions
svm = SVC(kernel='poly', degree=2)
svm.fit(xc, yc)
plot_decision_regions(xc, np.asarray(yc, dtype='int'), clf=svm, legend=2)
plt.xlabel('x1');plt.ylabel('x2')
plt.show()
```

- I used SVM (Support Vector Machines Classification) with a polynomial kernel in this classification to appropriately categorize the data.

NAME: ABHIK SARKAR
MATRIKELNUMMER: 23149662
IDM Id : lo88xide