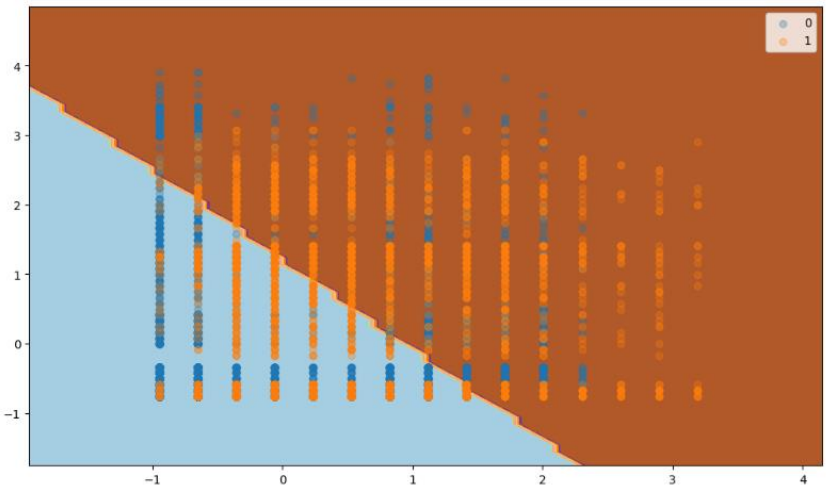
1. model_lr = LogisticRegression()
model_lr.fit(X_train_standardized, y_train)

2. model_l1 = LogisticRegression(penalty='l1', C=0.01, solver='saga', max_iter=1000)
model_l1.fit(X_train_standardized, y_train)

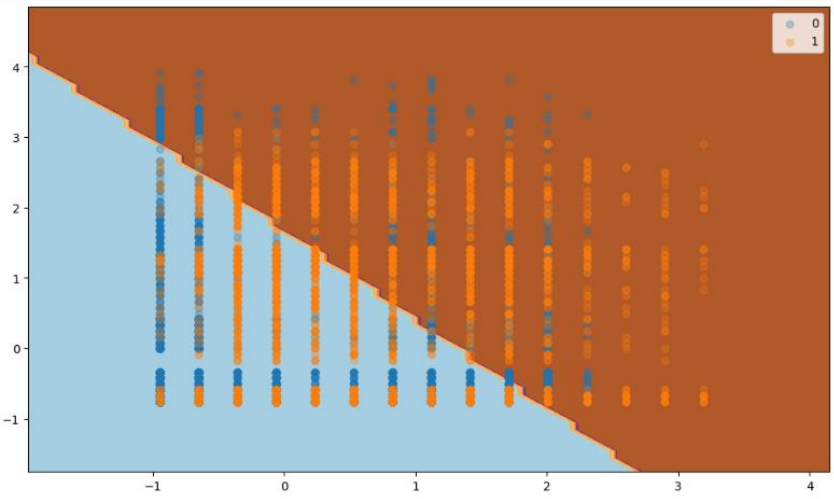
3. model_l2 = LogisticRegression(penalty='l2', C=0.001, solver='lbfgs', max_iter=1000)
model_l2.fit(X_train_standardized, y_train)

4. model_elasticnet = LogisticRegression(penalty='elasticnet', C=100, solver='saga', l1_ratio=0.5, max_iter=1000)
model_elasticnet.fit(X_train_standardized, y_train)

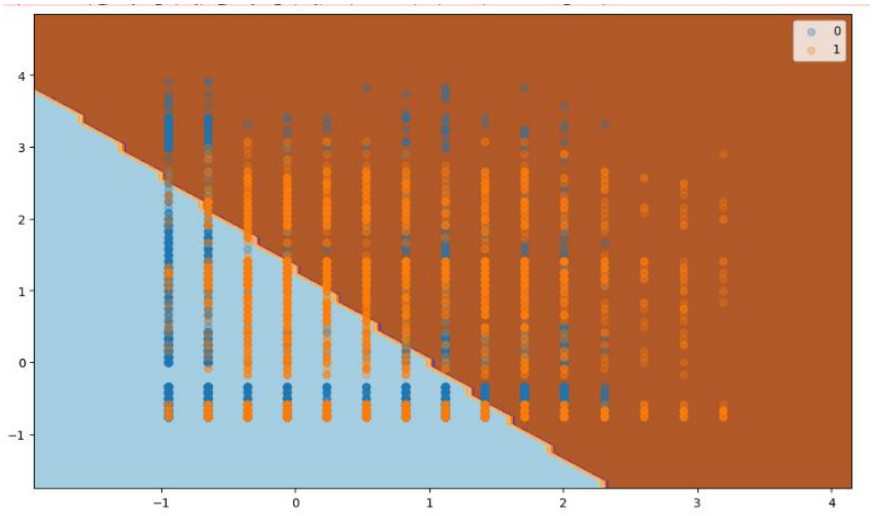
1.



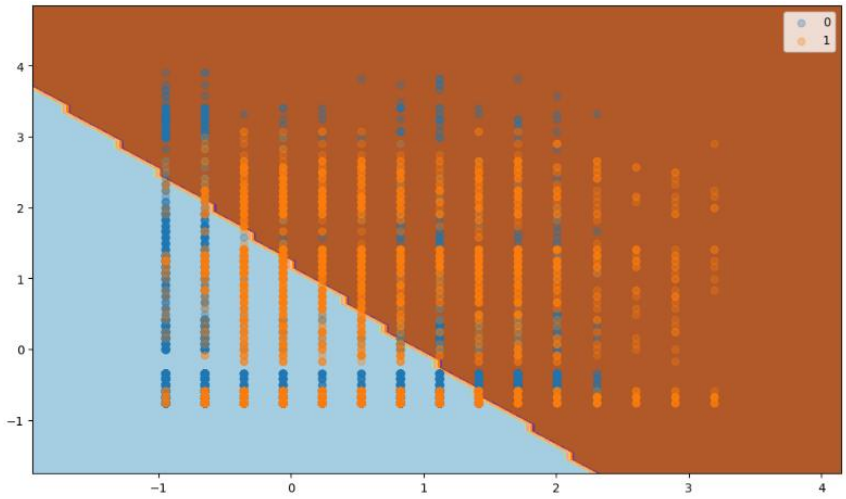
2.



3.



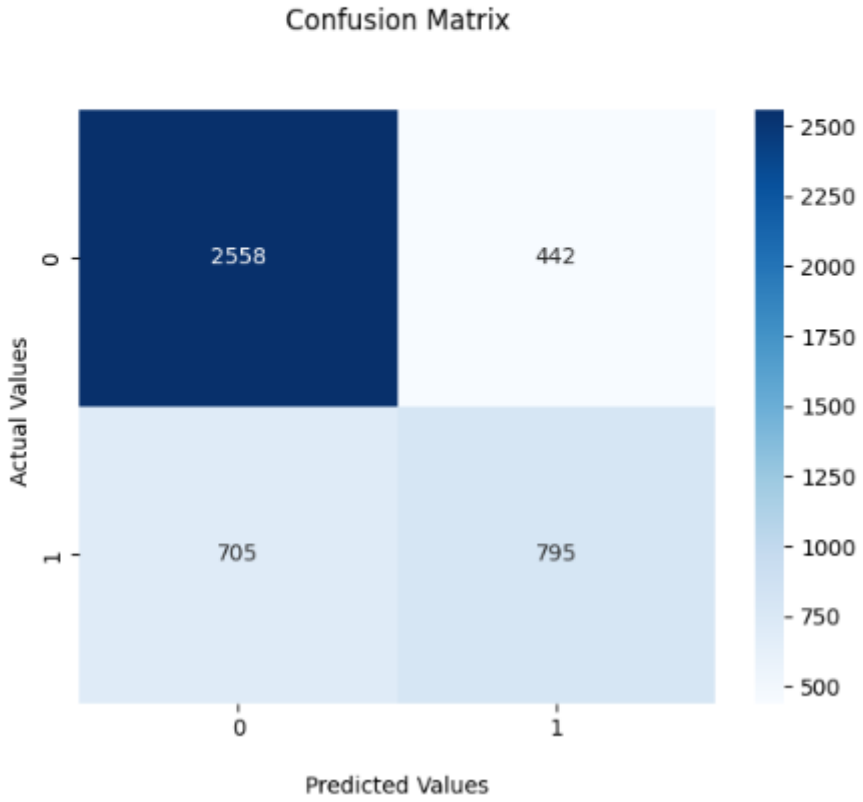
4.

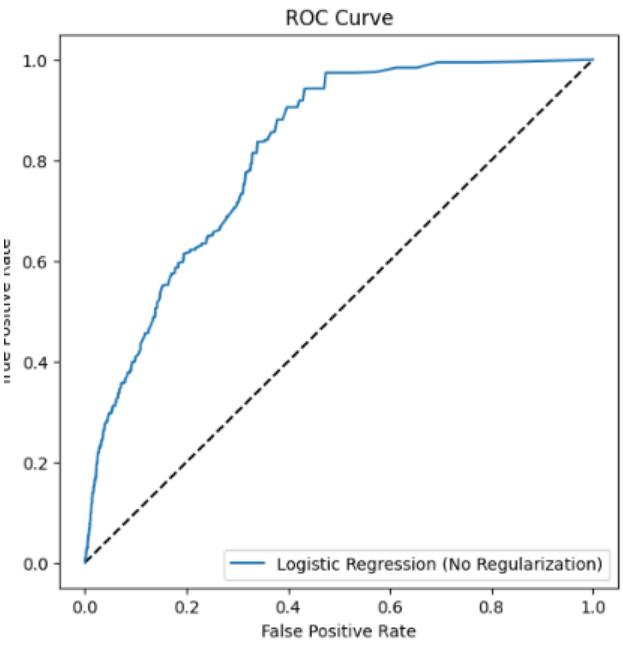


1.

Results for Logistic Regression (No Regularization):

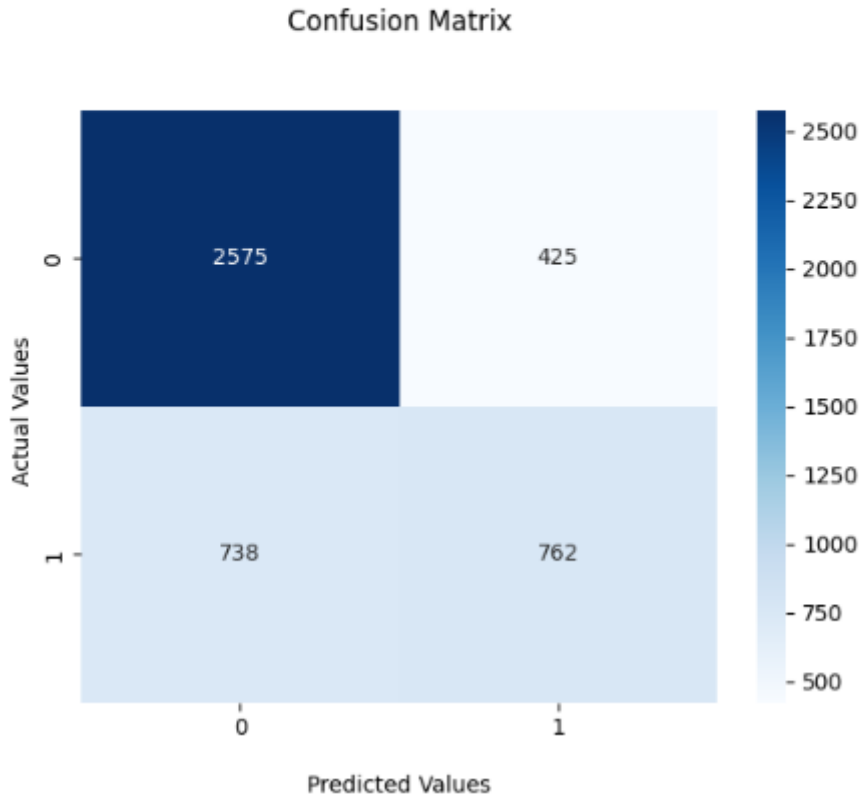
	precision	recall	f1-score	support
0	0.78	0.85	0.82	3000
1	0.64	0.53	0.58	1500
accuracy			0.75	4500
macro avg	0.71	0.69	0.70	4500
weighted avg	0.74	0.75	0.74	4500

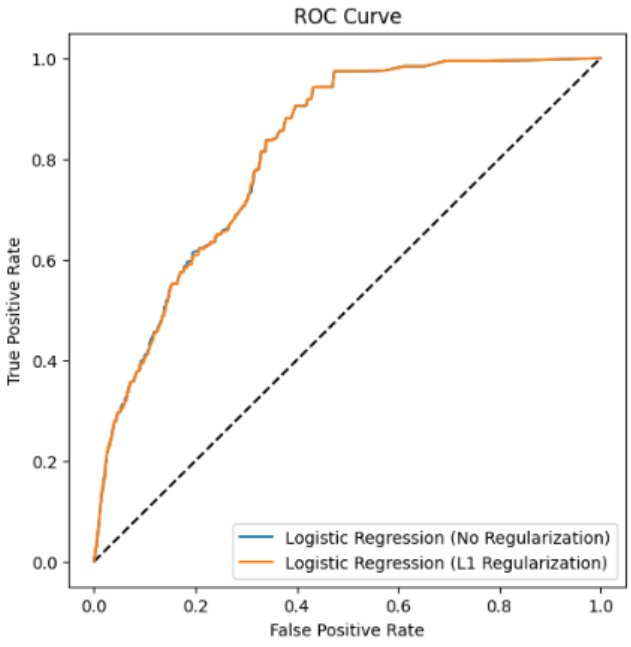




Results for Logistic Regression (L1 Regularization):

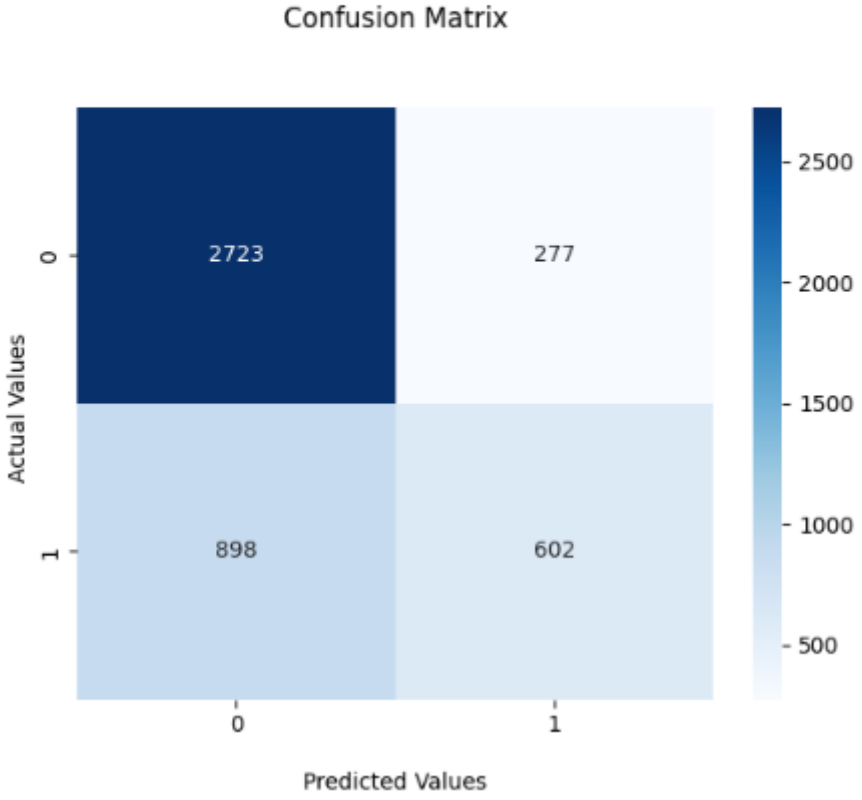
	precision	recall	f1-score	support
0	0.78	0.86	0.82	3000
1	0.64	0.51	0.57	1500
accuracy			0.74	4500
macro avg	0.71	0.68	0.69	4500
weighted avg	0.73	0.74	0.73	4500

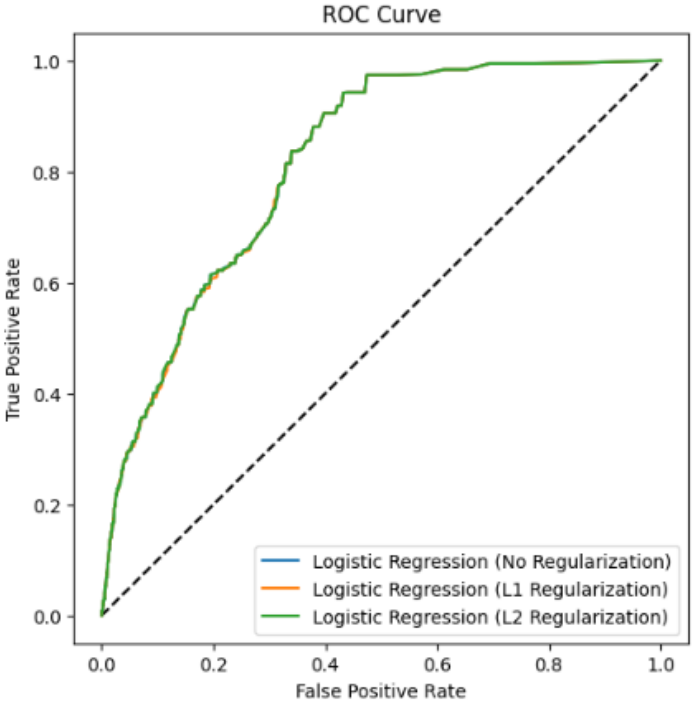




Results for Logistic Regression (L2 Regularization):

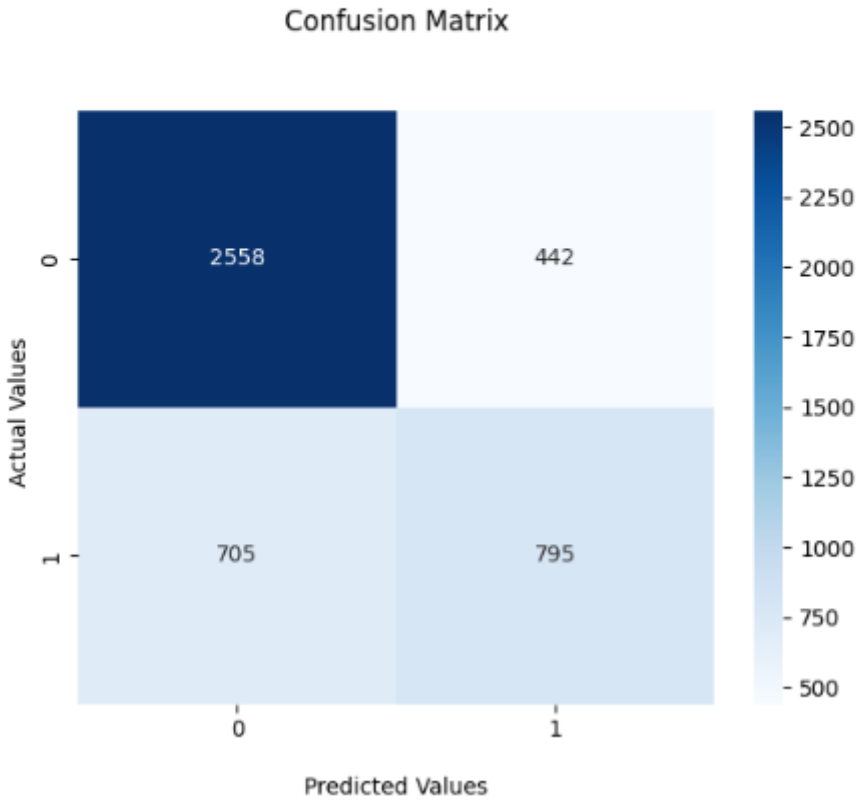
	precision	recall	f1-score	support
0	0.75	0.91	0.82	3000
1	0.68	0.40	0.51	1500
accuracy			0.74	4500
macro avg	0.72	0.65	0.66	4500
weighted avg	0.73	0.74	0.72	4500

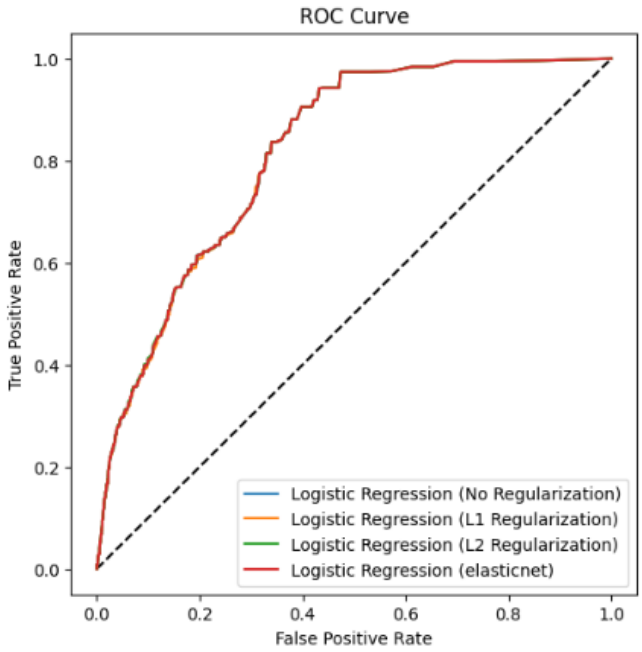




Results for Logistic Regression (elasticnet):

	precision	recall	f1-score	support
0	0.78	0.85	0.82	3000
1	0.64	0.53	0.58	1500
accuracy			0.75	4500
macro avg	0.71	0.69	0.70	4500
weighted avg	0.74	0.75	0.74	4500





	Model	F1_score	AUC
0	Logistic Regression (No Regularization)	0.580928	0.816493
1	Logistic Regression (L1 Regularization)	0.567175	0.816049
2	Logistic Regression (L2 Regularization)	0.506095	0.816606
3	Logistic Regression (elasticnet)	0.580928	0.816493

```
from sklearn.metrics import f1_score

# Obliczanie F1 score dla zbioru treningowego
y_train_pred = model_lr.predict(X_train_standardized)  # Predykcje na zbiorze treningowym
f1_train = f1_score(y_train, y_train_pred)  # F1 score na zbiorze treningowym
print(f"F1 score na zbiorze treningowym: {f1_train:.4f}")
```

```
F1 score na zbiorze treningowym: 0.5469
```

```
y_train_pred = model_l1.predict(X_train_standardized)  # Predykcje na zbiorze treningowym
f1_train = f1_score(y_train, y_train_pred)  # F1 score na zbiorze treningowym
print(f"F1 score na zbiorze treningowym: {f1_train:.4f}")
```

```
F1 score na zbiorze treningowym: 0.5353
```

```
y_train_pred = model_l2.predict(X_train_standardized)  # Predykcje na zbiorze treningowym
f1_train = f1_score(y_train, y_train_pred)  # F1 score na zbiorze treningowym
print(f"F1 score na zbiorze treningowym: {f1_train:.4f}")
```

```
F1 score na zbiorze treningowym: 0.4693
```

```
y_train_pred = model_elasticnet.predict(X_train_standardized)  # Predykcje na zbiorze treningowym
f1_train = f1_score(y_train, y_train_pred)  # F1 score na zbiorze treningowym
print(f"F1 score na zbiorze treningowym: {f1_train:.4f}")
```

```
F1 score na zbiorze treningowym: 0.5469
```