11 Supervised learning with logistic regression

Nandi Christmas

In [49]:
```python
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler
df_loan = pd.read_csv(
    filepath_or_buffer = "D:\\loan_data.csv",
    engine = 'pyarrow'
)
df_loan
```

Out[49]:

	loan_status	partition	person_education_2_moderate	person_education_3_high	person_home_ownership_3_high	loan_intent_2_moderate	loan_intent_3_high
0	1	test	0	0	1	1	0
1	0	test	1	0	0	0	0
2	1	train	1	0	0	0	1
3	1	train	0	1	1	0	1
4	1	test	0	0	1	0	1
...	...	...	...	...	...	...	...
44995	1	test	0	0	1	0	1
44996	1	validation	0	0	1	1	0
44997	1	train	0	0	1	0	1
44998	1	train	0	1	1	0	0
44999	1	train	1	0	1	0	1

45000 rows × 7 columns

In [50]:
```python
list_predictors = [
    'person_education_2_moderate',
        'person_education_3_high',
        'person_home_ownership_3_high',
        'loan_intent_2_moderate',
        'loan_intent_3_high',
]
```

Homework Request #1

In [51]:
```python
# Standardize the predictor variables
scaler = StandardScaler()
X = pd.DataFrame(scaler.fit_transform(X), columns=list_predictors)
```

In [52]:
```python
# Split data into training set
X_Train = df_loan.loc[df_loan['partition'] == 'train'][list_predictors]
y_Train = df_loan.loc[df_loan['partition'] == 'train', 'loan_status']
X_Train[list_predictors] = scaler.transform(X_Train[list_predictors])
```

Homework Question 2

In [53]:
```python
from sklearn.linear_model import LogisticRegression
list_LogisticRegression = [
    LogisticRegression(
        penalty = penalty,
        solver = 'saga',
        l1_ratio = 0.5, # ignored for non-elasticnet
        max_iter = 10000
    ) for penalty in [None,'l2','l1','elasticnet']
]
list_fit = [model.fit(
    X = X_Train,
    y = y_Train
) for model in list_LogisticRegression]
```

```
d:\python\Lib\site-packages\sklearn\linear_model\_logistic.py:1197: UserWarning: l1_ratio parameter is only used when penalty is 'elasticnet'. Got (penalty=None)
  warnings.warn(
d:\python\Lib\site-packages\sklearn\linear_model\_logistic.py:1197: UserWarning: l1_ratio parameter is only used when penalty is 'elasticnet'. Got (penalty=l2)
  warnings.warn(
d:\python\Lib\site-packages\sklearn\linear_model\_logistic.py:1197: UserWarning: l1_ratio parameter is only used when penalty is 'elasticnet'. Got (penalty=l1)
  warnings.warn(
```

In [54]:
```python
[fit.intercept_.round(1) for fit in list_fit]
```

Out[54]:
```
[array([-2.4]), array([-2.4]), array([-2.4]), array([-2.4])]
```

In [55]:
```python
[fit.coef_.round(1) for fit in list_fit]
```

Out[55]:
```
[array([[-0.1, -0. ,  1.4,  0.4,  0.7]]),
 array([[-0.1, -0. ,  1.4,  0.4,  0.7]]),
 array([[-0.1, -0. ,  1.4,  0.4,  0.7]]),
 array([[-0.1, -0. ,  1.4,  0.4,  0.7]])]
```

In [56]:
```python
list_predict = [fit.predict(X = X[list_predictors]) for fit in list_fit]
list_predict_probab = [fit.predict_proba(X = X[list_predictors])[:,1] for fit in list_fit]
df_predict_proba = pd.DataFrame(list_predict_probab).T
df_predict_proba
```

Out[56]:

	0	1	2	3
0	0.287672	0.287354	0.285956	0.286647
1	0.008033	0.008113	0.008142	0.008127
2	0.037255	0.037442	0.037468	0.037447
3	0.409555	0.409312	0.409634	0.409440
4	0.418099	0.417833	0.416483	0.417090
...	...	...	...	...
44995	0.418099	0.417833	0.416483	0.417090
44996	0.287672	0.287354	0.285956	0.286647
44997	0.418099	0.417833	0.416483	0.417090
44998	0.126763	0.127174	0.127643	0.127415
44999	0.373381	0.373259	0.373703	0.373400

45000 rows × 4 columns

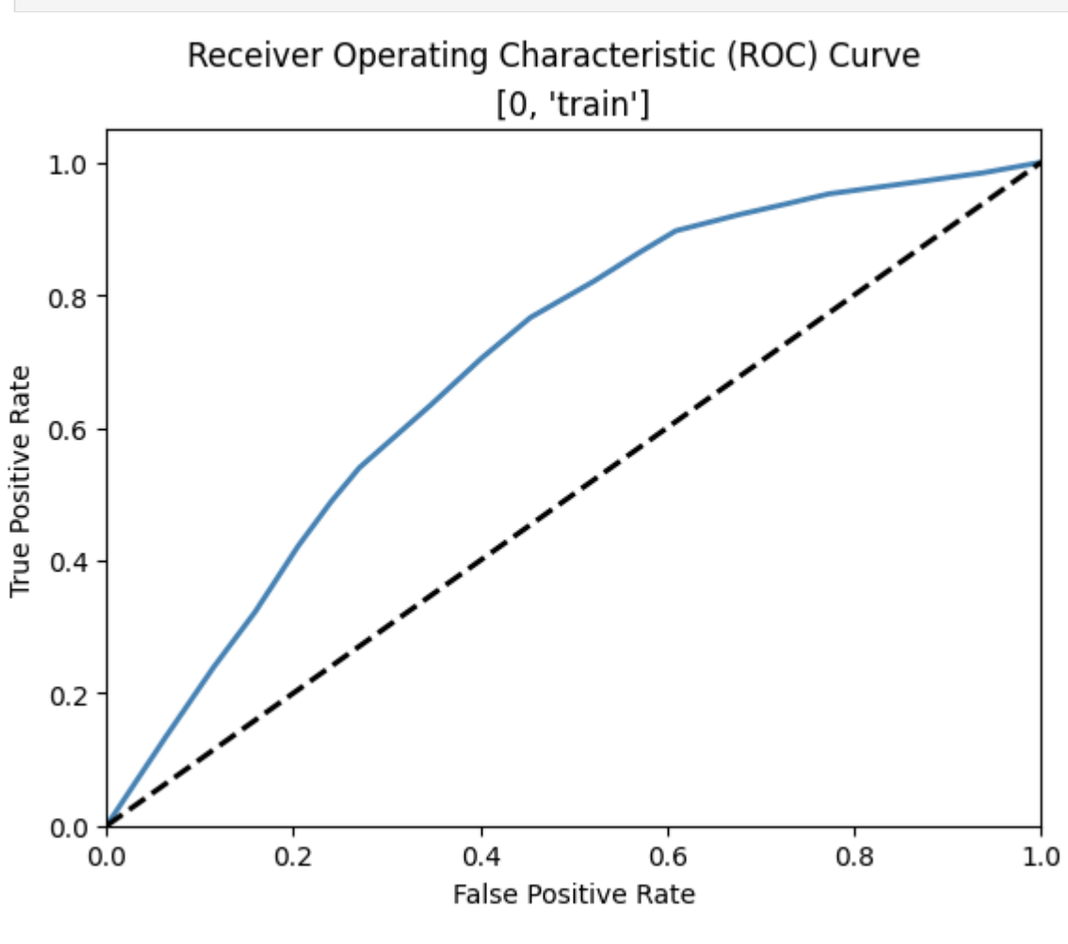
In [57]:
```python
from sklearn.metrics import roc_auc_score,roc_curve
df_loan['partition']
['train','validation','test']
list_roc_auc_score = [[j,k,roc_auc_score(
    y_true = df_loan.loc[df_loan['partition'] == k,'loan_status'],
    y_score = df_predict_proba.loc[df_loan['partition'] == k,j]
)] for j in range(len(list_predict_probab)) for k in ['train','validation','test']]
pd.DataFrame(list_roc_auc_score)
```

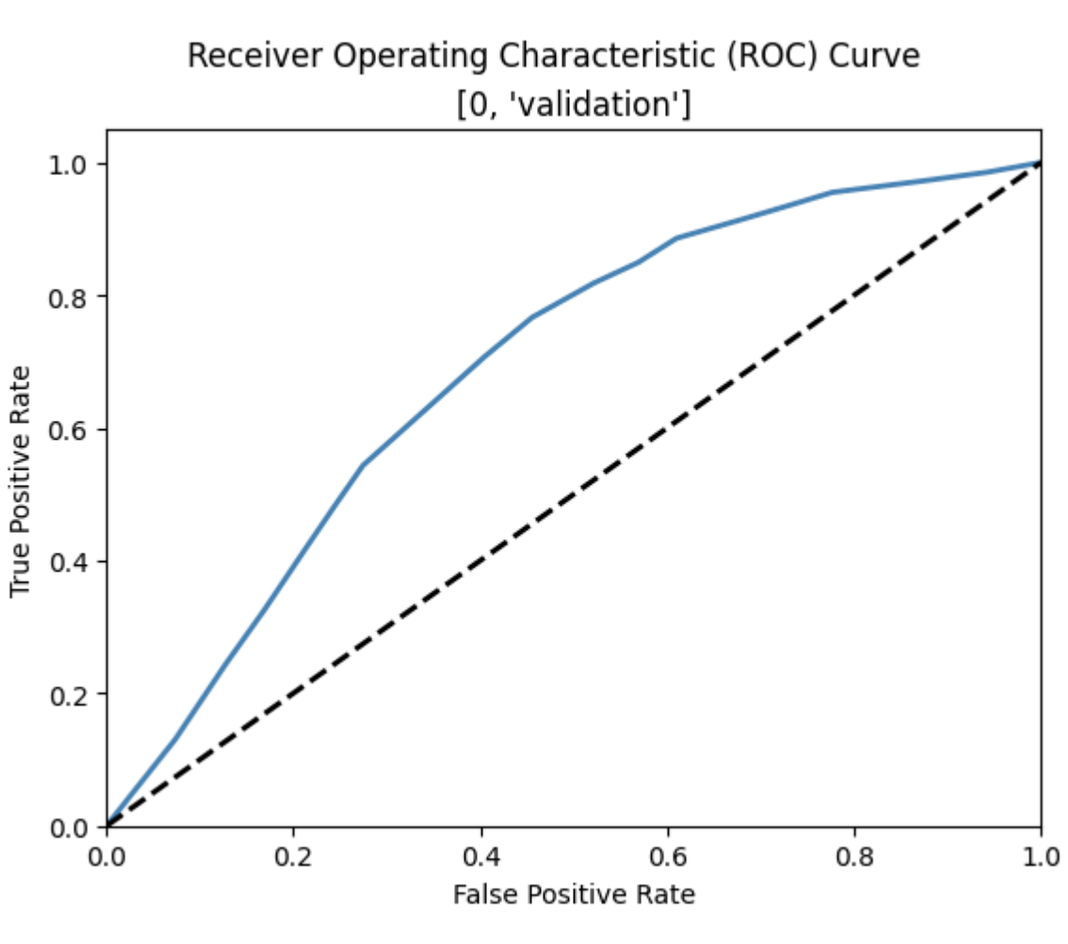
Out[57]:

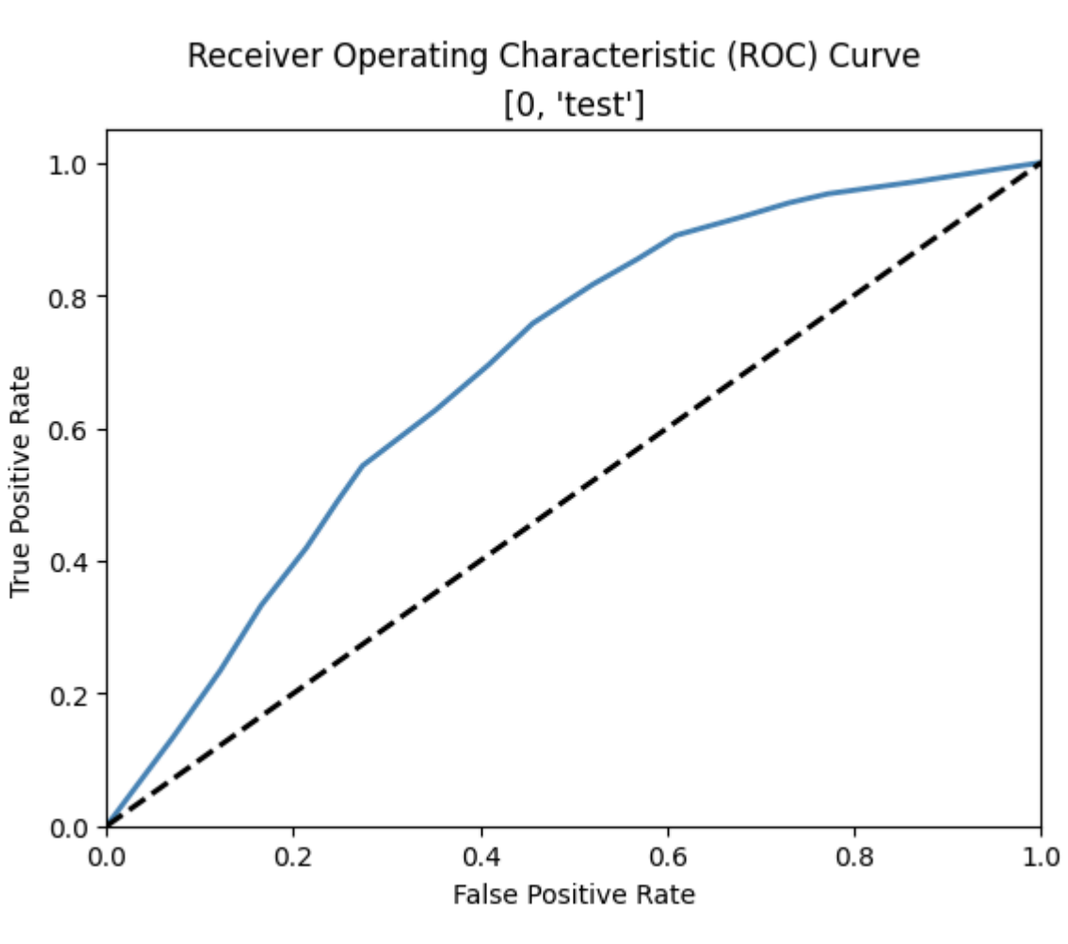
	0	1	2
0	0	train	0.696780
1	0	validation	0.690306
2	0	test	0.690289
3	1	train	0.696780
4	1	validation	0.690306
5	1	test	0.690289
6	2	train	0.696780
7	2	validation	0.690306
8	2	test	0.690289
9	3	train	0.696780
10	3	validation	0.690306
11	3	test	0.690289

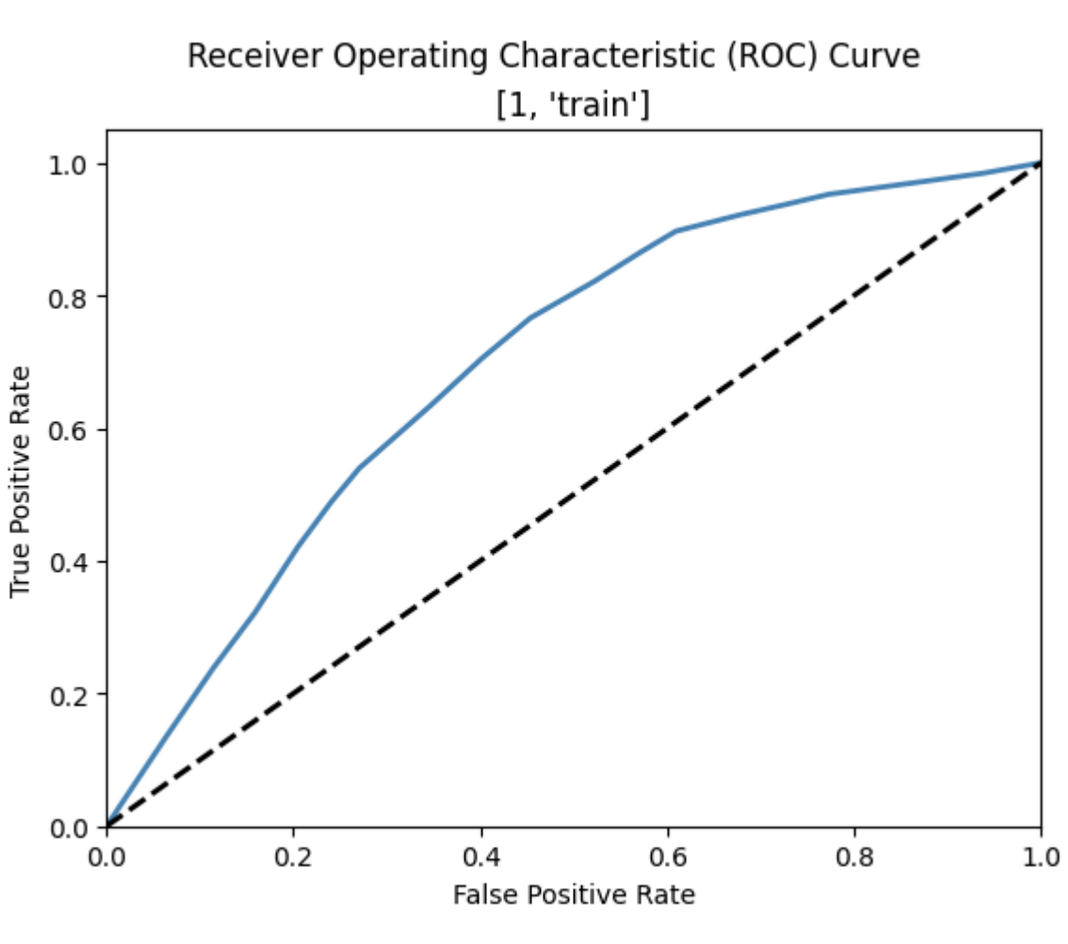
In [58]:
```python
list_roc_curve = [[j,k,roc_curve(
    y_true = df_loan.loc[df_loan['partition'] == k,'loan_status'],
    y_score = df_predict_proba.loc[df_loan['partition'] == k,j]
)] for j in range(len(list_predict_probab)) for k in ['train','validation','test']]
```

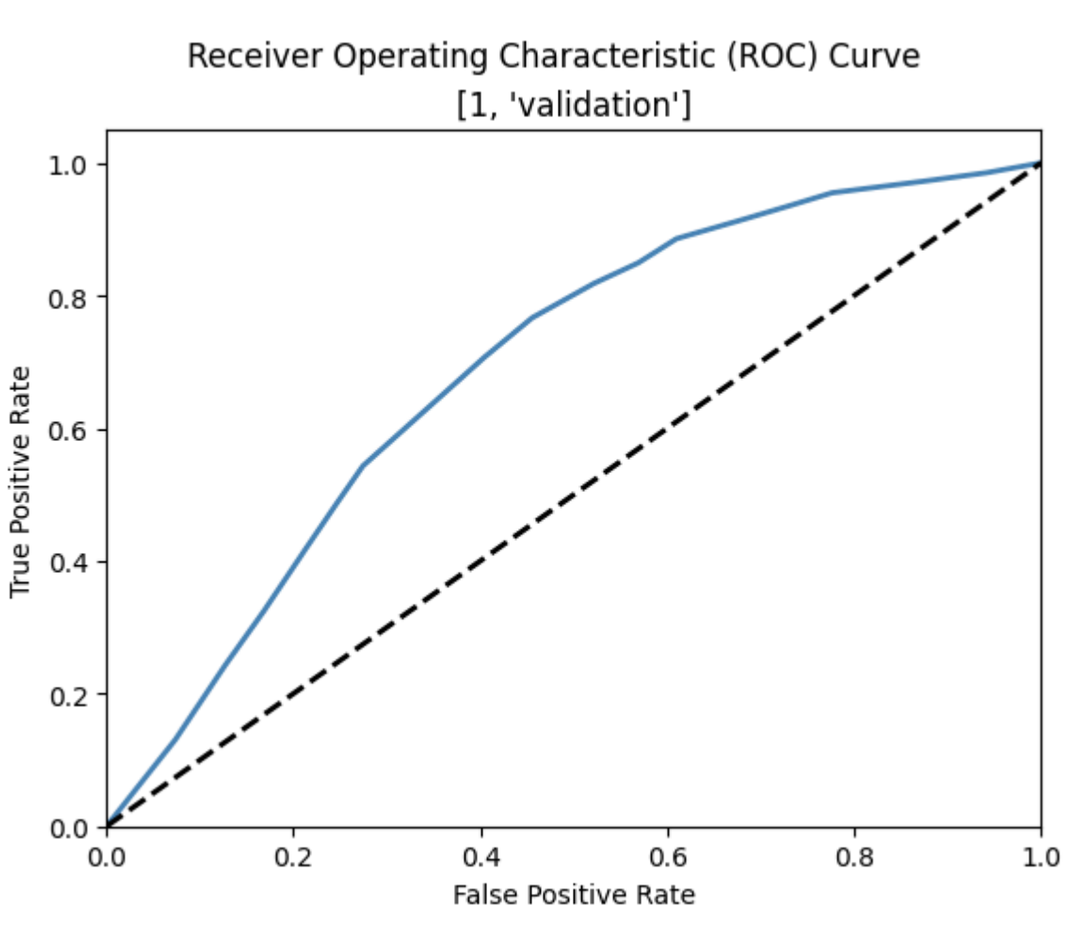
In [59]:
```python
import matplotlib.pyplot as plt
for j in range(len(list_roc_curve)):
    plt.figure()
    colors = ['steelblue']
    for i, color in zip(range(2), colors):
        plt.plot(
            list_roc_curve[j][2][0],
            list_roc_curve[j][2][1],
            color = color,
            lw = 2
        )
        plt.plot([0, 1], [0, 1], 'k--', lw=2)
        plt.xlim([0.0, 1.0])
        plt.ylim([0.0, 1.05])
        plt.xlabel('False Positive Rate')
        plt.ylabel('True Positive Rate')
        plt.suptitle('Receiver Operating Characteristic (ROC) Curve ')
        plt.title([list_roc_curve[j][0],list_roc_curve[j][1]])
        plt.show()

#fpr, tpr, thresholds = metrics.roc_curve(y, scores, pos_label=2)
```

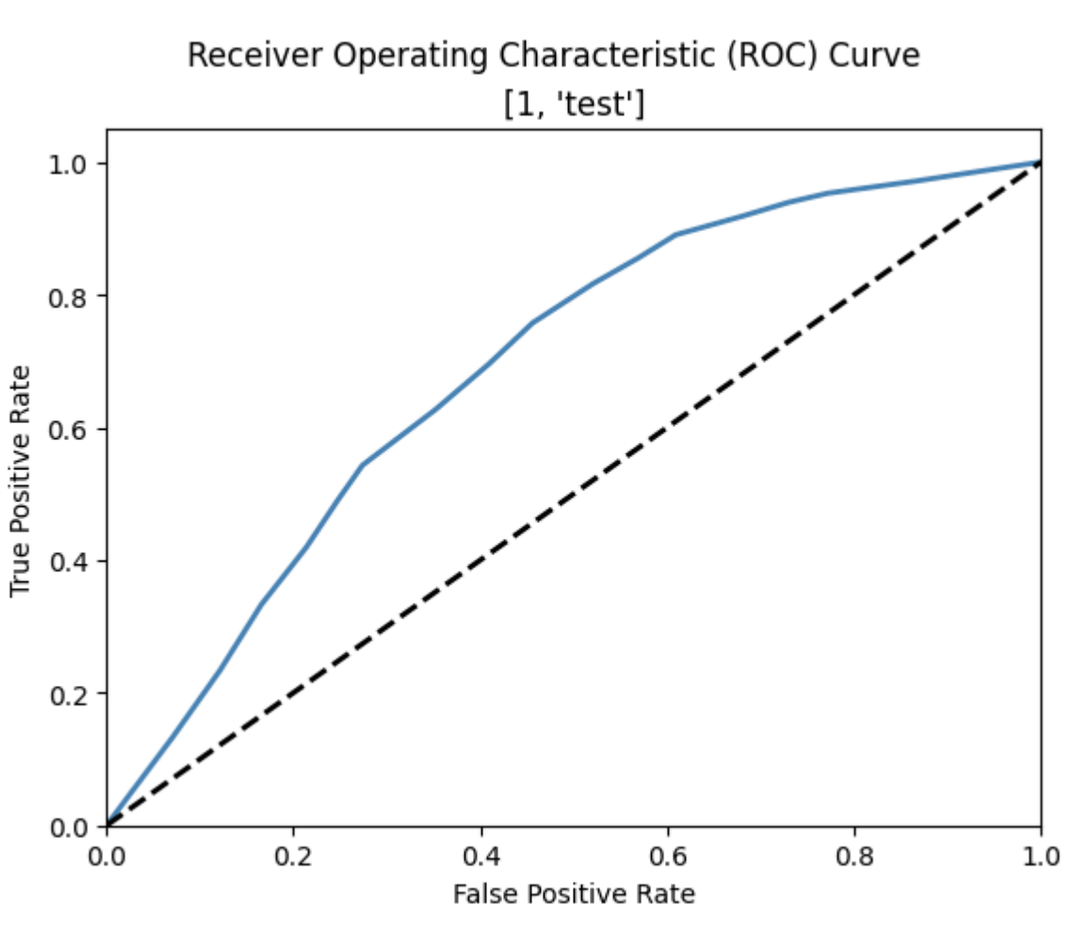


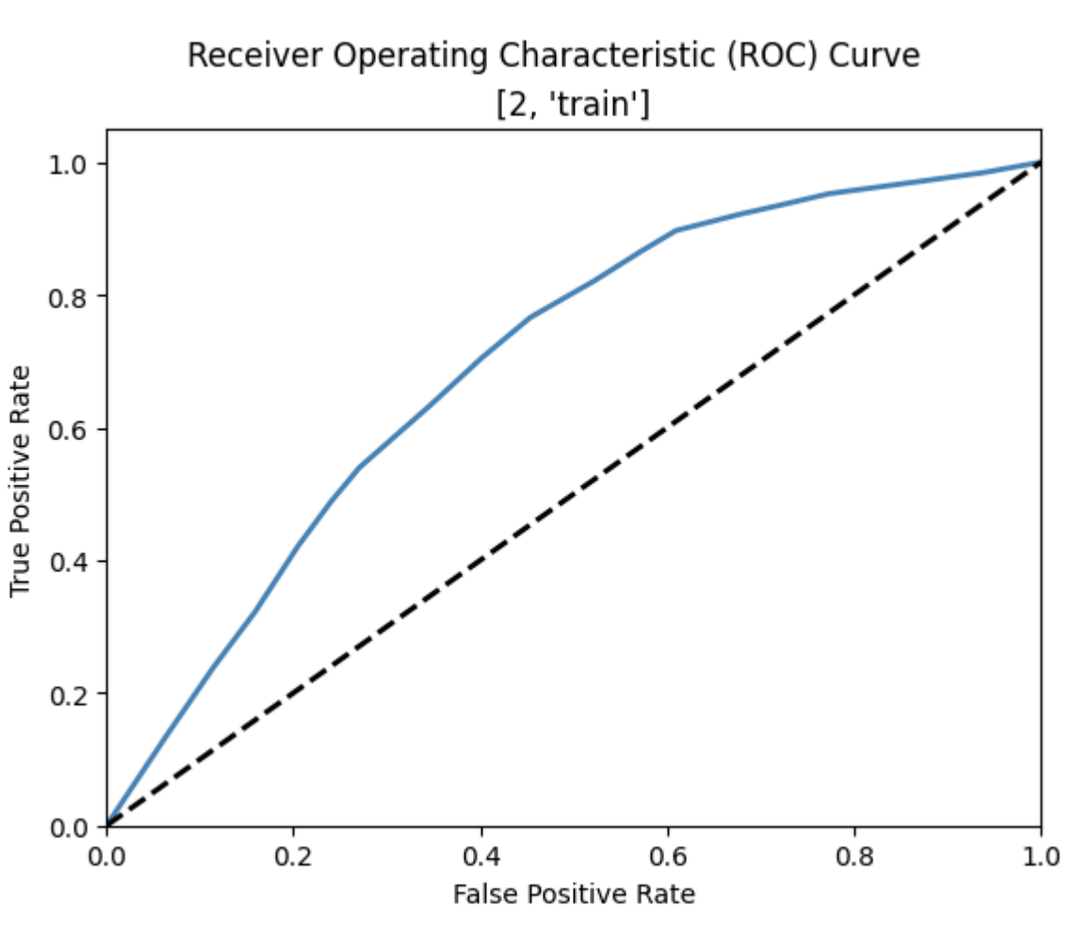


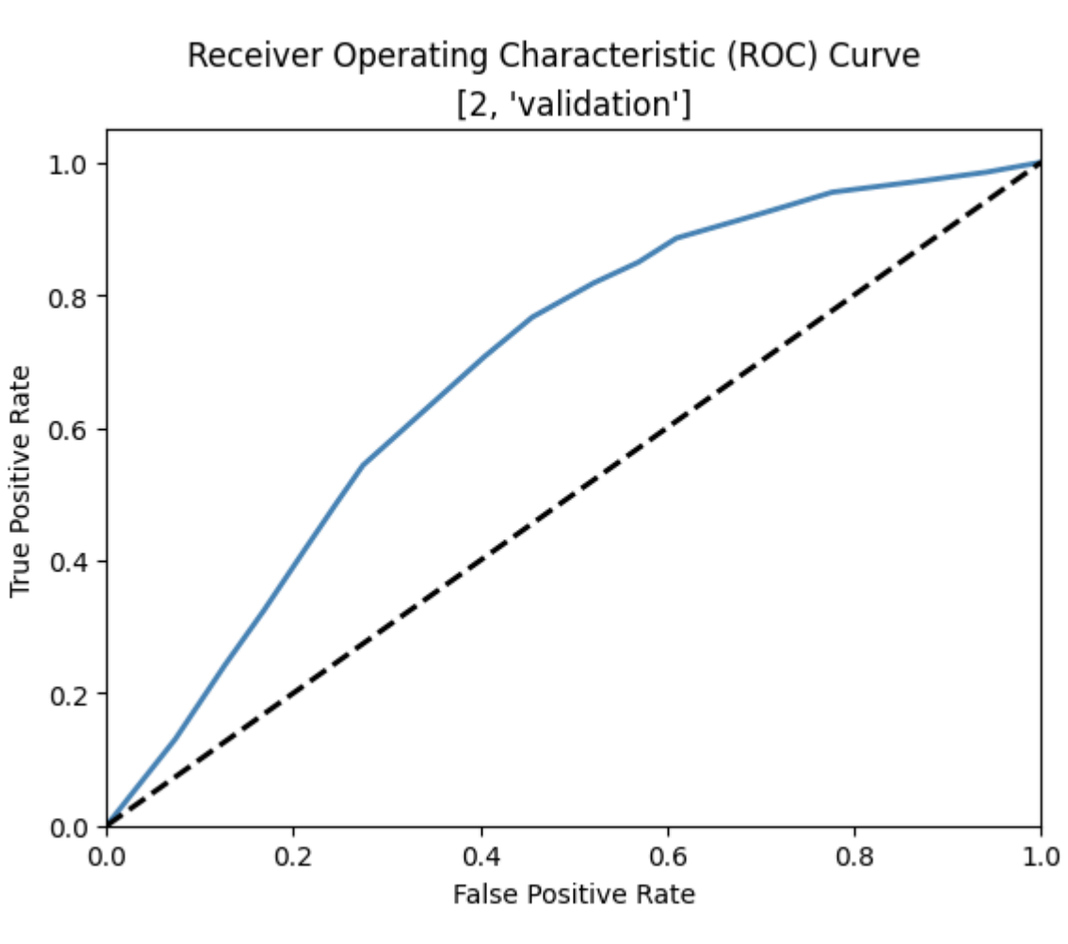


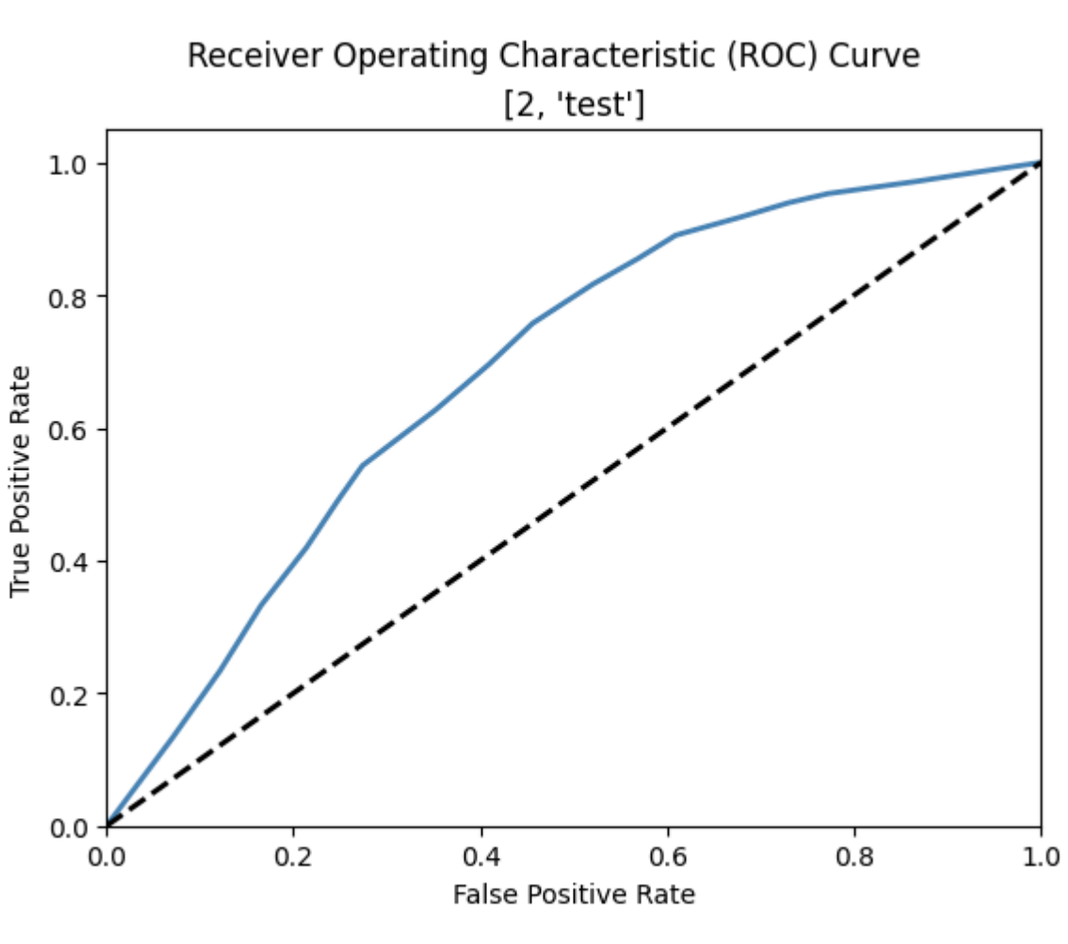


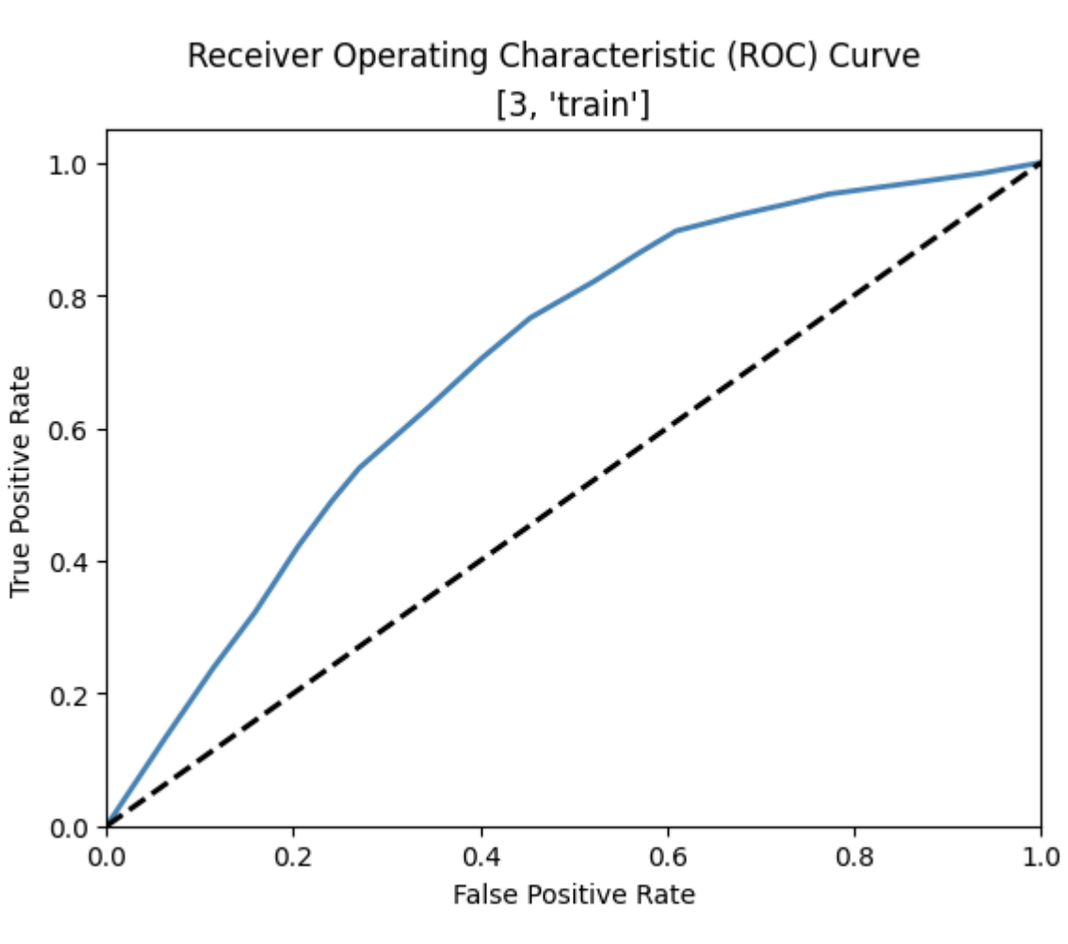


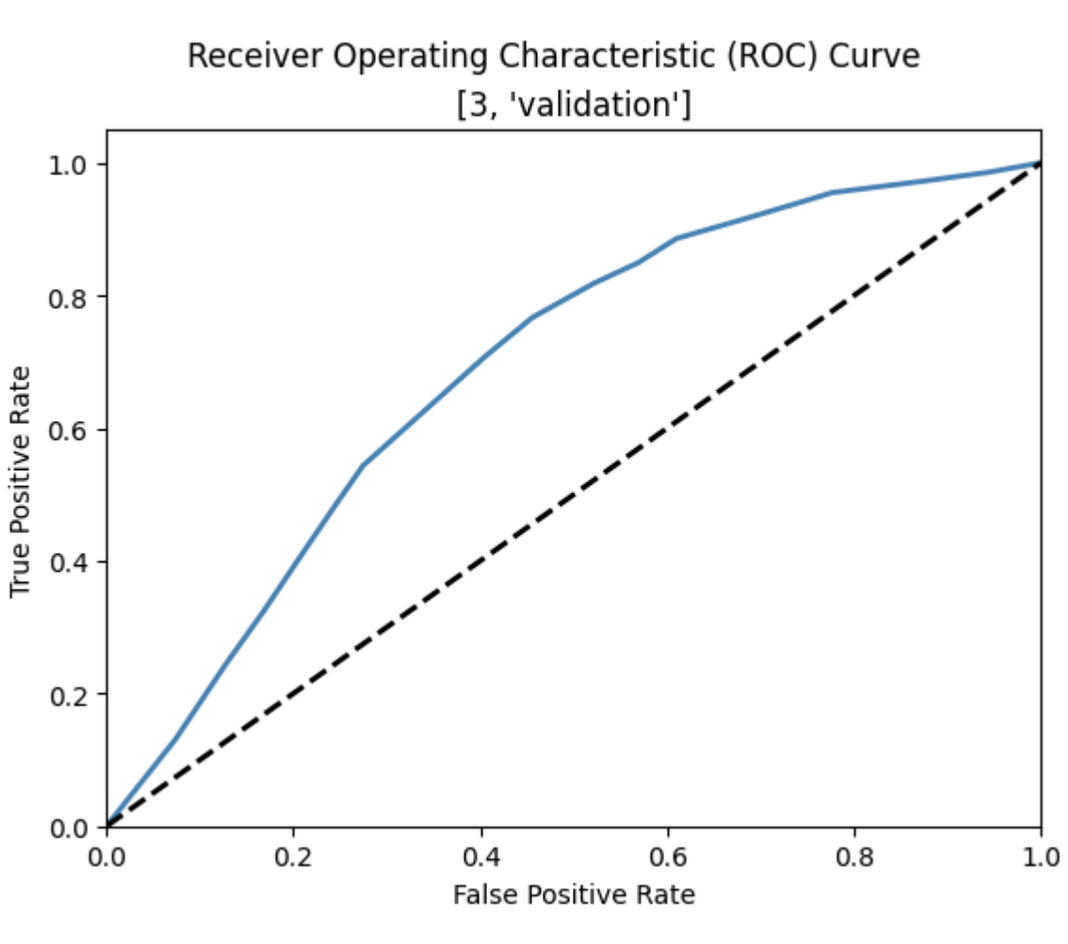


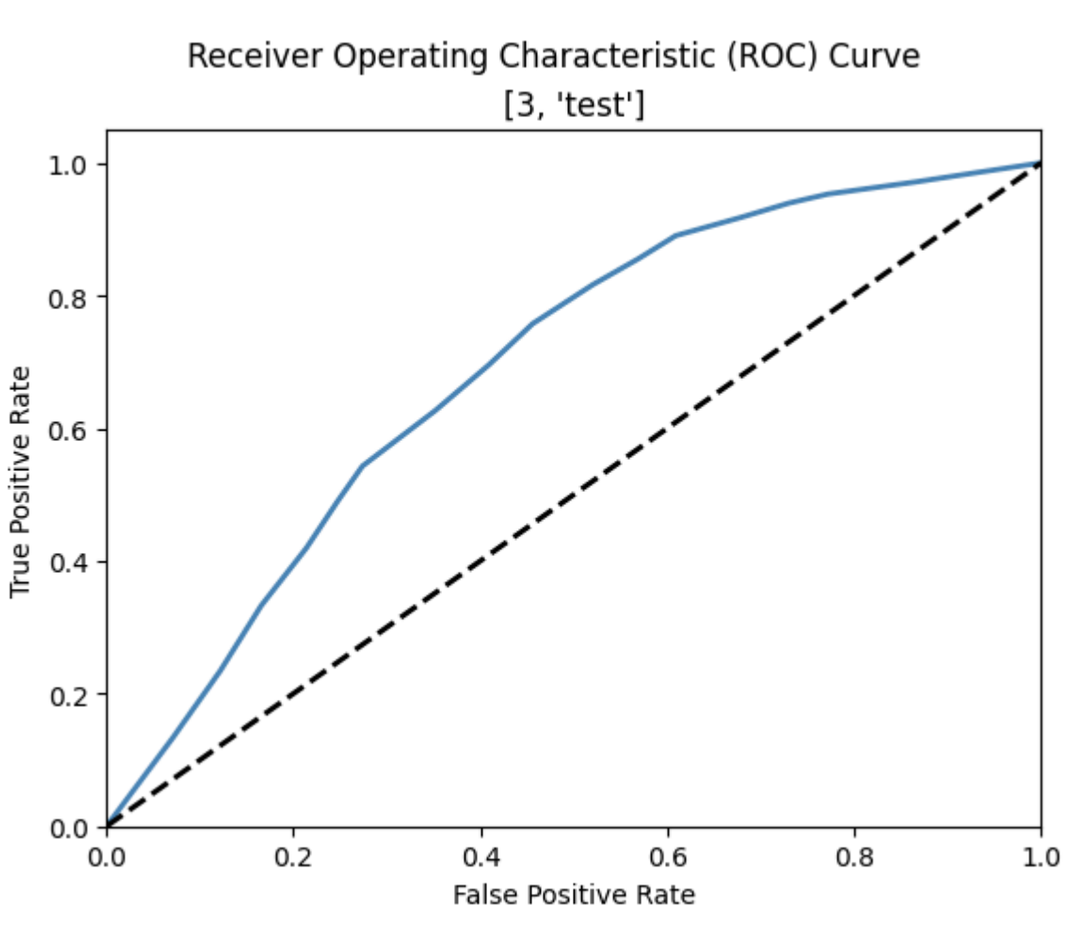








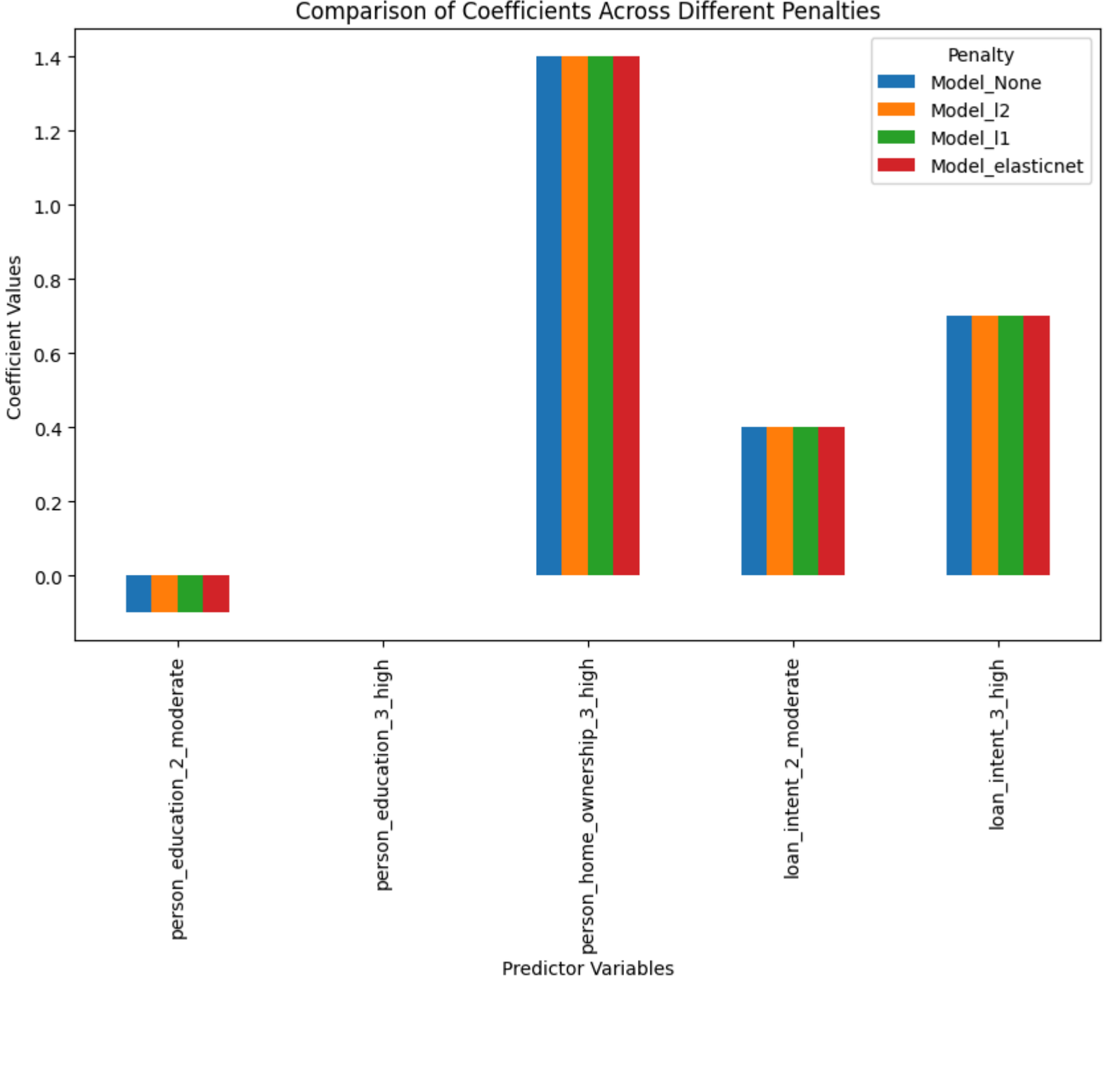




In [60]:
```python
# Display intercepts and coefficients
intercepts = [fit.intercept_.round(1) for fit in list_fit]
coefficients = [fit.coef_.round(1) for fit in list_fit]
print("Intercepts:", intercepts)
print("Coefficients:", coefficients)

# Create a DataFrame of coefficients for each model
coef_df = pd.DataFrame(
    {f'Model_{penalty}': coef.flatten() for penalty, coef in zip(['None', 'l2', 'l1', 'elasticnet'], coefficients)},
    index=list_predictors
)
```

```
Intercepts: [array([-2.4]), array([-2.4]), array([-2.4]), array([-2.4])]
Coefficients: [array([[-0.1, -0. ,  1.4,  0.4,  0.7]]), array([[-0.1, -0. ,  1.4,  0.4,  0.7]]), array([[-0.1, -0. ,  1.4,  0.4,  0.7]]), array([[-0.1, -0. ,  1.4,  0.4,  0.7]])]
```

In [61]:
```python
# Create a DataFrame of coefficients for each model
coef_df = pd.DataFrame(
    {f'Model_{penalty}': coef.flatten() for penalty, coef in zip(['None', 'l2', 'l1', 'elasticnet'], coefficients)},
    index=list_predictors
)

coef_df.plot(kind='bar', figsize=(10, 6))
plt.xlabel('Predictor Variables')
plt.ylabel('Coefficient Values')
plt.title('Comparison of Coefficients Across Different Penalties')
plt.legend(title='Penalty')
plt.show()
```



Homework Quesiton 3 NOTES ABOUT DIFFERENCE

It looks like model i2 and i1 have penalties that disallow overfitting. elsastionet has bot l1 and i2 in it. lastly, l1 is able to make the model ignore features, making the coefficient 0.