In [1]:

```
!pip install pytorch_tabnet
```

```
Collecting pytorch_tabnet
  Downloading pytorch_tabnet-4.1.0-py3-none-any.whl.metadata (15 kB)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.11/dist-packages (fr
om pytorch_tabnet) (2.0.2)
Requirement already satisfied: scikit_learn>0.21 in /usr/local/lib/python3.11/dist-packag
es (from pytorch_tabnet) (1.6.1)
Requirement already satisfied: scipy>1.4 in /usr/local/lib/python3.11/dist-packages (from
pytorch_tabnet) (1.15.3)
Requirement already satisfied: torch>=1.3 in /usr/local/lib/python3.11/dist-packages (fro
m pytorch_tabnet) (2.6.0+cu124)
Requirement already satisfied: tqdm>=4.36 in /usr/local/lib/python3.11/dist-packages (fro
m pytorch_tabnet) (4.67.1)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.11/dist-packages (
from scikit_learn>0.21->pytorch_tabnet) (1.5.1)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-pac
kages (from scikit_learn>0.21->pytorch_tabnet) (3.6.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (from
torch>=1.3->pytorch_tabnet) (3.18.0)
Requirement already satisfied: typing-extensions>=4.10.0 in /usr/local/lib/python3.11/dis
t-packages (from torch>=1.3->pytorch_tabnet) (4.14.0)
Requirement already satisfied: networkx in /usr/local/lib/python3.11/dist-packages (from
torch>=1.3->pytorch_tabnet) (3.5)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.11/dist-packages (from to
rch>=1.3->pytorch_tabnet) (3.1.6)
Requirement already satisfied: fsspec in /usr/local/lib/python3.11/dist-packages (from to
rch>=1.3->pytorch_tabnet) (2025.3.2)
Collecting nvidia-cuda-nvrtc-cu12==12.4.127 (from torch>=1.3->pytorch_tabnet)
  Downloading nvidia_cuda_nvrtc_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl.metadata
(1.5 kB)
Collecting nvidia-cuda-runtime-cu12==12.4.127 (from torch>=1.3->pytorch_tabnet)
  Downloading nvidia_cuda_runtime_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl.metadat
a (1.5 kB)
Collecting nvidia-cuda-cupti-cu12==12.4.127 (from torch>=1.3->pytorch_tabnet)
  Downloading nvidia_cuda_cupti_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl.metadata
(1.6 kB)
Collecting nvidia-cudnn-cu12==9.1.0.70 (from torch>=1.3->pytorch_tabnet)
  Downloading nvidia_cudnn_cu12-9.1.0.70-py3-none-manylinux2014_x86_64.whl.metadata (1.6
kB)
Collecting nvidia-cublas-cu12==12.4.5.8 (from torch>=1.3->pytorch_tabnet)
  Downloading nvidia_cublas_cu12-12.4.5.8-py3-none-manylinux2014_x86_64.whl.metadata (1.5
kB)
Collecting nvidia-cufft-cu12==11.2.1.3 (from torch>=1.3->pytorch_tabnet)
  Downloading nvidia_cufft_cu12-11.2.1.3-py3-none-manylinux2014_x86_64.whl.metadata (1.5
kB)
Collecting nvidia-curand-cu12==10.3.5.147 (from torch>=1.3->pytorch_tabnet)
  Downloading nvidia_curand_cu12-10.3.5.147-py3-none-manylinux2014_x86_64.whl.metadata (1
.5 kB)
Collecting nvidia-cusolver-cu12==11.6.1.9 (from torch>=1.3->pytorch_tabnet)
  Downloading nvidia_cusolver_cu12-11.6.1.9-py3-none-manylinux2014_x86_64.whl.metadata (1
.6 kB)
Collecting nvidia-cusparse-cu12==12.3.1.170 (from torch>=1.3->pytorch_tabnet)
  Downloading nvidia_cusparse_cu12-12.3.1.170-py3-none-manylinux2014_x86_64.whl.metadata
(1.6 kB)
Requirement already satisfied: nvidia-cusparselt-cu12==0.6.2 in /usr/local/lib/python3.11
/dist-packages (from torch>=1.3->pytorch_tabnet) (0.6.2)
Requirement already satisfied: nvidia-nccl-cu12==2.21.5 in /usr/local/lib/python3.11/dist
-packages (from torch>=1.3->pytorch_tabnet) (2.21.5)
Requirement already satisfied: nvidia-nvtx-cu12==12.4.127 in /usr/local/lib/python3.11/di
st-packages (from torch>=1.3->pytorch_tabnet) (12.4.127)
Collecting nvidia-nvjitlink-cu12==12.4.127 (from torch>=1.3->pytorch_tabnet)
  Downloading nvidia_nvjitlink_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl.metadata (
1.5 kB)
Requirement already satisfied: triton==3.2.0 in /usr/local/lib/python3.11/dist-packages (
from torch>=1.3->pytorch_tabnet) (3.2.0)
Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.11/dist-packages (
```

```
from torch>=1.3->pytorch_tabnet) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.11/dist-packa
ges (from sympy==1.13.1->torch>=1.3->pytorch_tabnet) (1.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.11/dist-packages
(from jinja2->torch>=1.3->pytorch_tabnet) (3.0.2)
Downloading pytorch_tabnet-4.1.0-py3-none-any.whl (44 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 44.5/44.5 kB 1.4 MB/s eta 0:00:00
Downloading nvidia_cublas_cu12-12.4.5.8-py3-none-manylinux2014_x86_64.whl (363.4 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 363.4/363.4 MB 4.4 MB/s eta 0:00:00
Downloading nvidia_cuda_cupti_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl (13.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.8/13.8 MB 95.9 MB/s eta 0:00:00
Downloading nvidia_cuda_nvrtc_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl (24.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 24.6/24.6 MB 74.8 MB/s eta 0:00:00
Downloading nvidia_cuda_runtime_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl (883 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 883.7/883.7 kB 48.5 MB/s eta 0:00:00
Downloading nvidia_cudnn_cu12-9.1.0.70-py3-none-manylinux2014_x86_64.whl (664.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 664.8/664.8 MB 2.1 MB/s eta 0:00:00
Downloading nvidia_cufft_cu12-11.2.1.3-py3-none-manylinux2014_x86_64.whl (211.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 211.5/211.5 MB 5.4 MB/s eta 0:00:00
Downloading nvidia_curand_cu12-10.3.5.147-py3-none-manylinux2014_x86_64.whl (56.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 56.3/56.3 MB 11.7 MB/s eta 0:00:00
Downloading nvidia_cusolver_cu12-11.6.1.9-py3-none-manylinux2014_x86_64.whl (127.9 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 127.9/127.9 MB 7.5 MB/s eta 0:00:00
Downloading nvidia_cusparse_cu12-12.3.1.170-py3-none-manylinux2014_x86_64.whl (207.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 207.5/207.5 MB 6.1 MB/s eta 0:00:00
Downloading nvidia_nvjitlink_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl (21.1 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 21.1/21.1 MB 83.9 MB/s eta 0:00:00
Installing collected packages: nvidia-nvjitlink-cu12, nvidia-curand-cu12, nvidia-cufft-cu
12, nvidia-cuda-runtime-cu12, nvidia-cuda-nvrtc-cu12, nvidia-cuda-cupti-cu12, nvidia-cubl
as-cu12, nvidia-cusparse-cu12, nvidia-cudnn-cu12, nvidia-cusolver-cu12, pytorch_tabnet
  Attempting uninstall: nvidia-nvjitlink-cu12
    Found existing installation: nvidia-nvjitlink-cu12 12.5.82
    Uninstalling nvidia-nvjitlink-cu12-12.5.82:
      Successfully uninstalled nvidia-nvjitlink-cu12-12.5.82
  Attempting uninstall: nvidia-curand-cu12
    Found existing installation: nvidia-curand-cu12 10.3.6.82
    Uninstalling nvidia-curand-cu12-10.3.6.82:
      Successfully uninstalled nvidia-curand-cu12-10.3.6.82
  Attempting uninstall: nvidia-cufft-cu12
    Found existing installation: nvidia-cufft-cu12 11.2.3.61
    Uninstalling nvidia-cufft-cu12-11.2.3.61:
      Successfully uninstalled nvidia-cufft-cu12-11.2.3.61
  Attempting uninstall: nvidia-cuda-runtime-cu12
    Found existing installation: nvidia-cuda-runtime-cu12 12.5.82
    Uninstalling nvidia-cuda-runtime-cu12-12.5.82:
      Successfully uninstalled nvidia-cuda-runtime-cu12-12.5.82
  Attempting uninstall: nvidia-cuda-nvrtc-cu12
    Found existing installation: nvidia-cuda-nvrtc-cu12 12.5.82
    Uninstalling nvidia-cuda-nvrtc-cu12-12.5.82:
      Successfully uninstalled nvidia-cuda-nvrtc-cu12-12.5.82
  Attempting uninstall: nvidia-cuda-cupti-cu12
    Found existing installation: nvidia-cuda-cupti-cu12 12.5.82
    Uninstalling nvidia-cuda-cupti-cu12-12.5.82:
      Successfully uninstalled nvidia-cuda-cupti-cu12-12.5.82
  Attempting uninstall: nvidia-cublas-cu12
    Found existing installation: nvidia-cublas-cu12 12.5.3.2
    Uninstalling nvidia-cublas-cu12-12.5.3.2:
      Successfully uninstalled nvidia-cublas-cu12-12.5.3.2
  Attempting uninstall: nvidia-cusparse-cu12
    Found existing installation: nvidia-cusparse-cu12 12.5.1.3
    Uninstalling nvidia-cusparse-cu12-12.5.1.3:
      Successfully uninstalled nvidia-cusparse-cu12-12.5.1.3
  Attempting uninstall: nvidia-cudnn-cu12
    Found existing installation: nvidia-cudnn-cu12 9.3.0.75
    Uninstalling nvidia-cudnn-cu12-9.3.0.75:
      Successfully uninstalled nvidia-cudnn-cu12-9.3.0.75
  Attempting uninstall: nvidia-cusolver-cu12
    Found existing installation: nvidia-cusolver-cu12 11.6.3.83
    Uninstalling nvidia-cusolver-cu12-11.6.3.83:
      Successfully uninstalled nvidia-cusolver-cu12-11.6.3.83
Successfully installed nvidia-cublas-cu12-12.4.5.8 nvidia-cuda-cupti-cu12-12.4.127 nvidia
-cuda-nvrtc-cu12-12.4.127 nvidia-cuda-runtime-cu12-12.4.127 nvidia-cudnn-cu12-9.1.0.70 nv
```

```
idia-cufft-cu12-11.2.1.3 nvidia-curand-cu12-10.3.5.147 nvidia-cusolver-cu12-11.6.1.9 nvid
ia-cusparse-cu12-12.3.1.170 nvidia-nvjitlink-cu12-12.4.127 pytorch_tabnet-4.1.0
```

In [2]:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import StratifiedKFold, train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_score, roc_
auc_score
from pytorch_tabnet.tab_model import TabNetClassifier
import torch
from sklearn.metrics import roc_curve, auc
```

In [3]:

```
# Loading dataset
df = pd.read_csv("heart_disease_uci.csv")
df.drop(columns=["id", "dataset"], inplace=True)
df["target"] = df["num"].apply(lambda x: 1 if x > 0 else 0)
df.drop(columns=["num"], inplace=True)

numerical = ['age', 'trestbps', 'chol', 'thalch', 'oldpeak', 'ca']
categorical = ['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'thal']
```

In [4]:

```
# Handling missing values and encoding categorical features
df[numerical] = SimpleImputer(strategy="median").fit_transform(df[numerical])
for col in categorical:
    df[col] = LabelEncoder().fit_transform(df[col].astype(str))
df[numerical] = StandardScaler().fit_transform(df[numerical])

X = df[numerical + categorical].values
y = df["target"].values
```

In [5]:

```
# Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, test_size=0.2, ran
dom_state=42)

# TabNet Classifier
tabnet = TabNetClassifier(
    n_d=64, n_a=64, n_steps=5,
    gamma=1.5, n_independent=2, n_shared=2,
    momentum=0.3,
    lambda_sparse=1e-4,
    optimizer_fn=torch.optim.Adam,
    optimizer_params=dict(lr=2e-2),
    scheduler_params={"step_size":10, "gamma":0.9},
    scheduler_fn=torch.optim.lr_scheduler.StepLR,
    verbose=0,
    seed=42
)
```

In [6]:

```
tabnet.fit(
    X_train, y_train,
    eval_set=[(X_train, y_train), (X_test, y_test)],
    eval_name=['train', 'val'],
    eval_metric=['accuracy'],
    max_epochs=200,
    patience=20,
    batch_size=256,
    virtual_batch_size=128,
    num_workers=0,
    drop_last=False
```

```
)
```

```
Early stopping occurred at epoch 53 with best_epoch = 33 and best_val_accuracy = 0.83152
```

```
/usr/local/lib/python3.11/dist-packages/pytorch_tabnet/callbacks.py:172: UserWarning: Bes
t weights from best epoch are automatically used!
  warnings.warn(wrn_msg)
```

In [7]:

```
# Predictions
y_pred_proba = tabnet.predict_proba(X_test)[:, 1]
y_pred = (y_pred_proba > 0.5).astype(int)
```

In [8]:

```
# Metrics
results = {
    'Model': "TabNet",
    'Accuracy': accuracy_score(y_test, y_pred),
    'F1 Score': f1_score(y_test, y_pred),
    'Precision': precision_score(y_test, y_pred),
    'Recall': recall_score(y_test, y_pred),
    'ROC AUC': roc_auc_score(y_test, y_pred_proba)
}

print("\n Final Evaluation Metrics for TabNet:")
for k, v in results.items():
    print(f"{k}: {v:.4f}" if isinstance(v, float) else f"{k}: {v}")
```

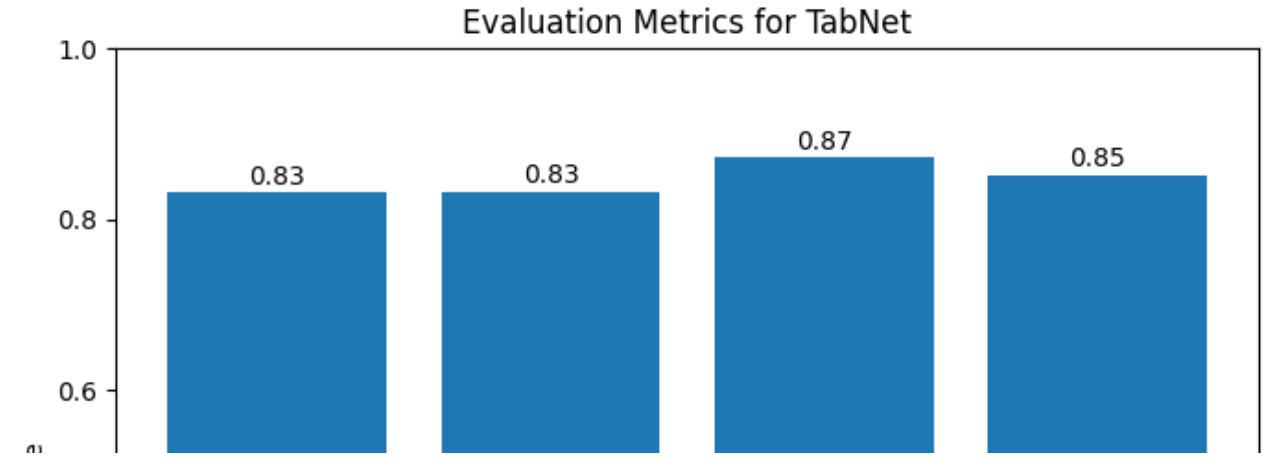
```
 Final Evaluation Metrics for TabNet:
Model: TabNet
Accuracy: 0.8315
F1 Score: 0.8517
Precision: 0.8318
Recall: 0.8725
ROC AUC: 0.8733
```

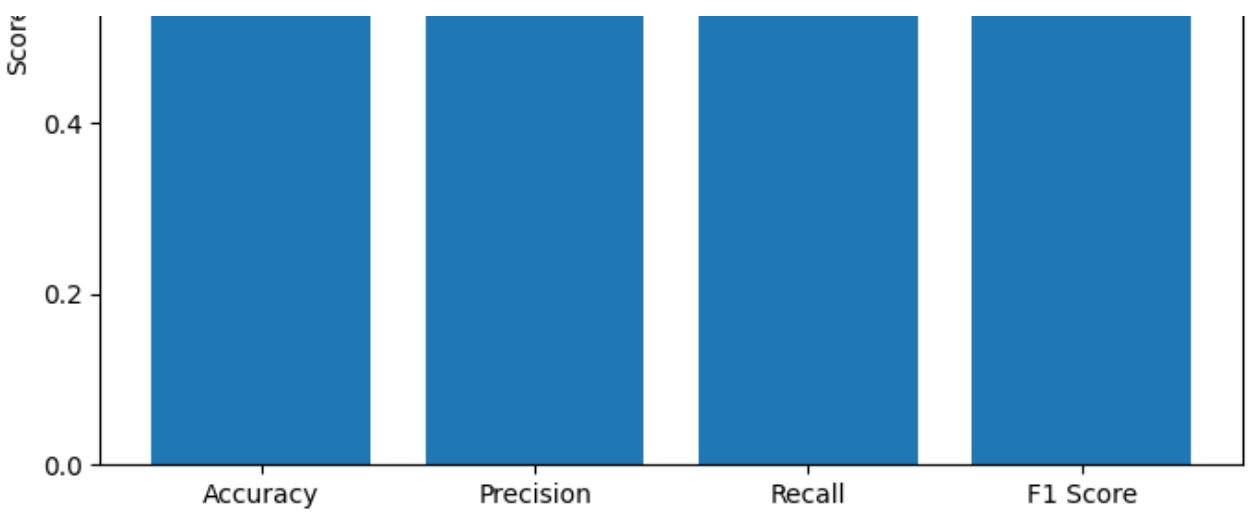
In [9]:

```
import matplotlib.pyplot as plt
```

In [10]:

```
# bar plot
metrics_to_plot = ['Accuracy', 'Precision', 'Recall', 'F1 Score']
scores = [results[m] for m in metrics_to_plot]

plt.figure(figsize=(8, 6))
plt.bar(metrics_to_plot, scores)
plt.ylabel("Score")
plt.title("Evaluation Metrics for TabNet")
plt.ylim(0, 1)
for i, v in enumerate(scores):
    plt.text(i, v + 0.01, f"{v:.2f}", ha='center', fontsize=10)
plt.show()
```





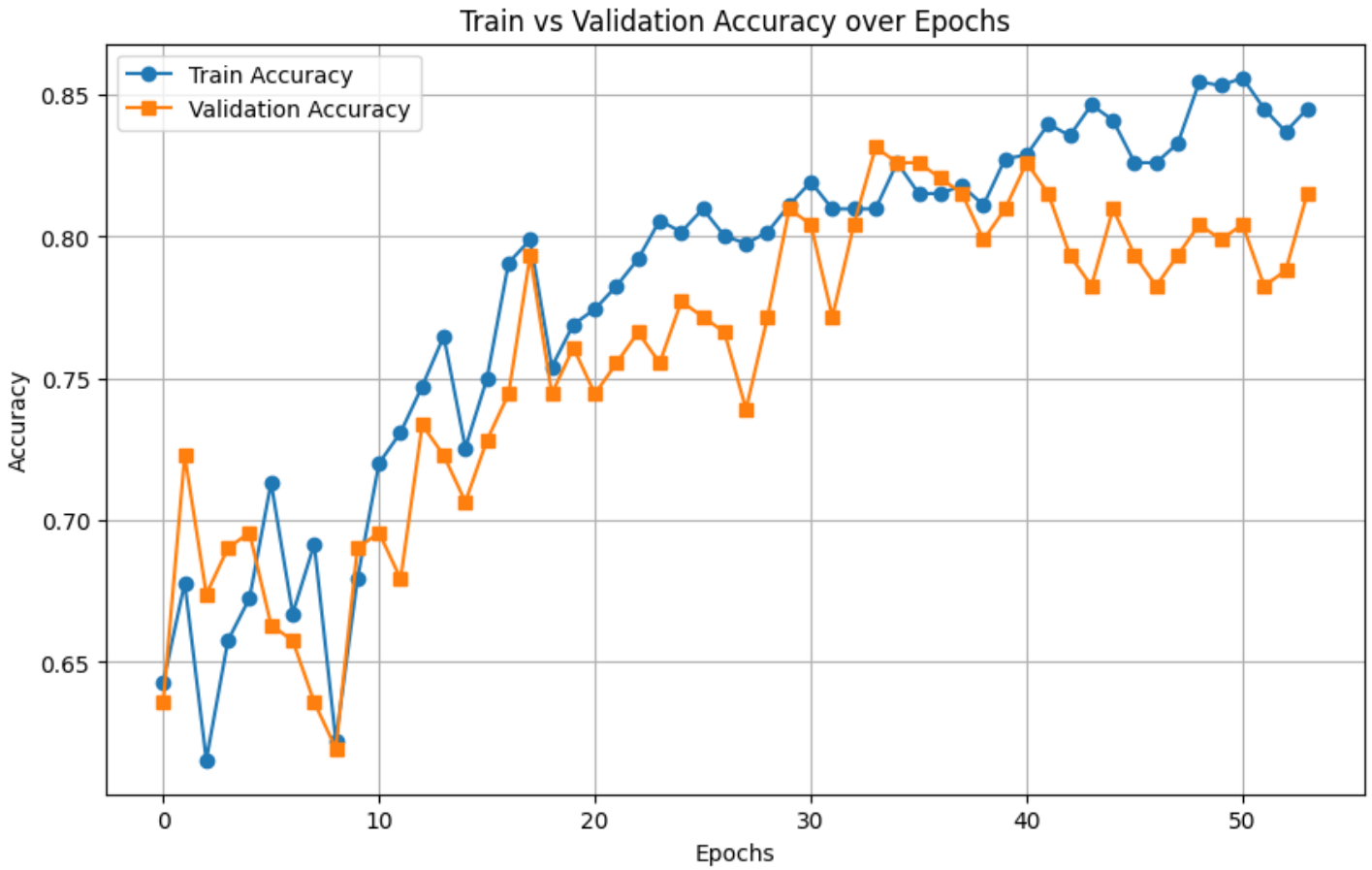
In [11]:

```
print(tabnet.history)
```

{'loss': [2.4532782098521357, 0.929909436599068, 0.875175442384637, 0.697170483029407, 0.6802718665288843, 0.6767560191776442, 0.5367364378079124, 0.522400122621785, 0.5099219524342081, 0.5920395877050317, 0.5276923659055129, 0.5025692582130432, 0.5084480202716329, 0.4902825251869533, 0.45624422897463257, 0.45727975990461267, 0.43445225254349085, 0.4362872424332992, 0.40345087777013366, 0.4106170275936956, 0.4067475562510283, 0.41243676258170087, 0.416572075823079, 0.39114161937133124, 0.3933981151684471, 0.4106549711331077, 0.38139555894810223, 0.3797016882378122, 0.3894627612570058, 0.3856390986753547, 0.36661087041315826, 0.3610478833965633, 0.371664179408032, 0.36346577950145886, 0.3613346918769505, 0.35077390463455865, 0.3582947383756223, 0.3572672242703645, 0.3677983037803484, 0.34862256957137067, 0.3493110511613929, 0.32485075489334436, 0.35788060012071027, 0.34521216801975085, 0.36768279386603314, 0.32397001074708026, 0.32349352732948633, 0.33373622661051544, 0.31024884141009784, 0.322559416294097 9, 0.33066059325052344, 0.3242557243160579, 0.33626394686491595, 0.3400524336358775], 'lr': [0.02, 0.02, 0.02, 0.02, 0.02, 0.02, 0.02, 0.02, 0.02, 0.02, 0.018000000000000002, 0.018000000000000002, 0.018000000000000002, 0.018000000000000002, 0.018000000000000002, 0.018000000000000002, 0.018000000000000002, 0.018000000000000002, 0.018000000000000002, 0.016200000000000003, 0.016200000000000003, 0.016200000000000003, 0.016200000000000003, 0.016200000000000003, 0.016200000000000003, 0.016200000000000003, 0.016200000000000003, 0.014580000000000003, 0.014580000000000003, 0.014580000000000003, 0.014580000000000003, 0.014580000000000003, 0.014580000000000003, 0.014580000000000003, 0.014580000000000003, 0.013122000000000003, 0.013122000000000003, 0.013122000000000003, 0.013122000000000003, 0.013122000000000003, 0.013122000000000003, 0.013122000000000003, 0.013122000000000003, 0.011809800000000004, 0.011809800000000004, 0.011809800000000004, 0.011809800000000004], 'train_accuracy': [0.6426630434782609, 0.6779891304347826, 0.6154891304347826, 0.657608695652174, 0.6725543478260869, 0.7133152173913043, 0.6671195652173914, 0.6915760869565217, 0.6222826086956522, 0.6793478260869565, 0.720108695652174, 0.7309782608695652, 0.7472826086956522, 0.7649456521739131, 0.7255434782608695, 0.75, 0.7907608695652174, 0.7989130434782609, 0.7540760869565217, 0.7690217391304348, 0.7744565217391305, 0.782608695652174, 0.7921195652173914, 0.8057065217391305, 0.8016304347826086, 0.8097826086956522, 0.8002717391304348, 0.7975543478260869, 0.8016304347826086, 0.811141304347826, 0.8192934782608695, 0.8097826086956522, 0.8097826086956522, 0.8097826086956522, 0.8260869565217391, 0.8152173913043478, 0.8152173913043478, 0.8179347826086957, 0.811141304347826, 0.8274456521739131, 0.8288043478260869, 0.8396739130434783, 0.8355978260869565, 0.8464673913043478, 0.8410326086956522, 0.8260869565217391, 0.8260869565217391, 0.8328804347826086, 0.8546195652173914, 0.8532608695652174, 0.8559782608695652, 0.845108695652174, 0.8369565217391305, 0.8451086956521741], 'val_accuracy': [0.6358695652173914, 0.7228260869565217, 0.6739130434782609, 0.6902173913043478, 0.6956521739130435, 0.6630434782608695, 0.657608695652174, 0.6358695652173914, 0.6195652173913043, 0.6902173913043478, 0.6956521739130435, 0.6793478260869565, 0.7336956521739131, 0.7228260869565217, 0.7065217391304348, 0.7282608695652174, 0.7445652173913043, 0.7934782608695652, 0.7445652173913043, 0.7608695652173914, 0.7445652173913043, 0.7554347826086957, 0.7663043478260869, 0.7554347826086957, 0.7771739130434783, 0.7717391304347826, 0.7663043478260869, 0.7391304347826086, 0.7717391304347826, 0.8097826086956522, 0.8043478260869565, 0.7717391304347826, 0.8043478260869565, 0.8315217391304348, 0.8260869565217391, 0.8260869565217391, 0.8206521739130435, 0.8152173913043478, 0.7989130434782609, 0.8097826086956522, 0.8260869565217391, 0.8152173913043478, 0.7934782608695652, 0.782608695652174, 0.8097826086956522, 0.7934782608695652, 0.782608695652174, 0.7934782608695652, 0.8043478260869565, 0.7989130434782609, 0.8043478260869565, 0.782608695652174, 0.7880434782608695, 0.8152173913043478]}

In [12]:

```
# Extract both train and validation accuracy
train_acc = tabnet.history['train_accuracy']
val_acc = tabnet.history['val_accuracy']

# Plot
plt.figure(figsize=(10, 6))
plt.plot(train_acc, label='Train Accuracy', marker='o')
plt.plot(val_acc, label='Validation Accuracy', marker='s')
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.title("Train vs Validation Accuracy over Epochs")
plt.legend()
plt.grid(True)
plt.show()
```



In [13]:

```
from pytorch_tabnet.tab_model import TabNetClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_
auc_score
import matplotlib.pyplot as plt
```

In [14]:

```
# Defining the regularized TabNet model
tabnet = TabNetClassifier(
    n_d=32, n_a=32,
    n_steps=5,
    gamma=1.5,
    n_independent=2, n_shared=2,
    lambda_sparse=1e-3,          # Regularization
    momentum=0.5,                # Smoothing
    optimizer_fn=torch.optim.Adam,
    optimizer_params=dict(lr=0.01),
    scheduler_params={"step_size": 10, "gamma": 0.9},
    scheduler_fn=torch.optim.lr_scheduler.StepLR,
    seed=42,
    verbose=0
)
```

In [15]:

```
# Fitting model with both train and validation sets to track accuracy
tabnet.fit(
    X_train, y_train,
    eval_set=[(X_train, y_train), (X_test, y_test)],
    eval_name=['train', 'val'],
    eval_metric=['accuracy'],
    max_epochs=200,
    patience=20,
    batch_size=256,
    virtual_batch_size=128,
    num_workers=0,
    drop_last=False
)
```

```
Early stopping occurred at epoch 70 with best_epoch = 50 and best_val_accuracy = 0.84239
```

```
/usr/local/lib/python3.11/dist-packages/pytorch_tabnet/callbacks.py:172: UserWarning: Bes
t weights from best epoch are automatically used!
  warnings.warn(wrn_msg)
```

In [16]:

```
# Predicting and calculating metrics
y_pred_proba = tabnet.predict_proba(X_test)[:, 1]
y_pred = (y_pred_proba > 0.5).astype(int)

results = {
    'Accuracy': accuracy_score(y_test, y_pred),
    'Precision': precision_score(y_test, y_pred),
    'Recall': recall_score(y_test, y_pred),
    'F1 Score': f1_score(y_test, y_pred),
    'ROC AUC': roc_auc_score(y_test, y_pred_proba)
}
```

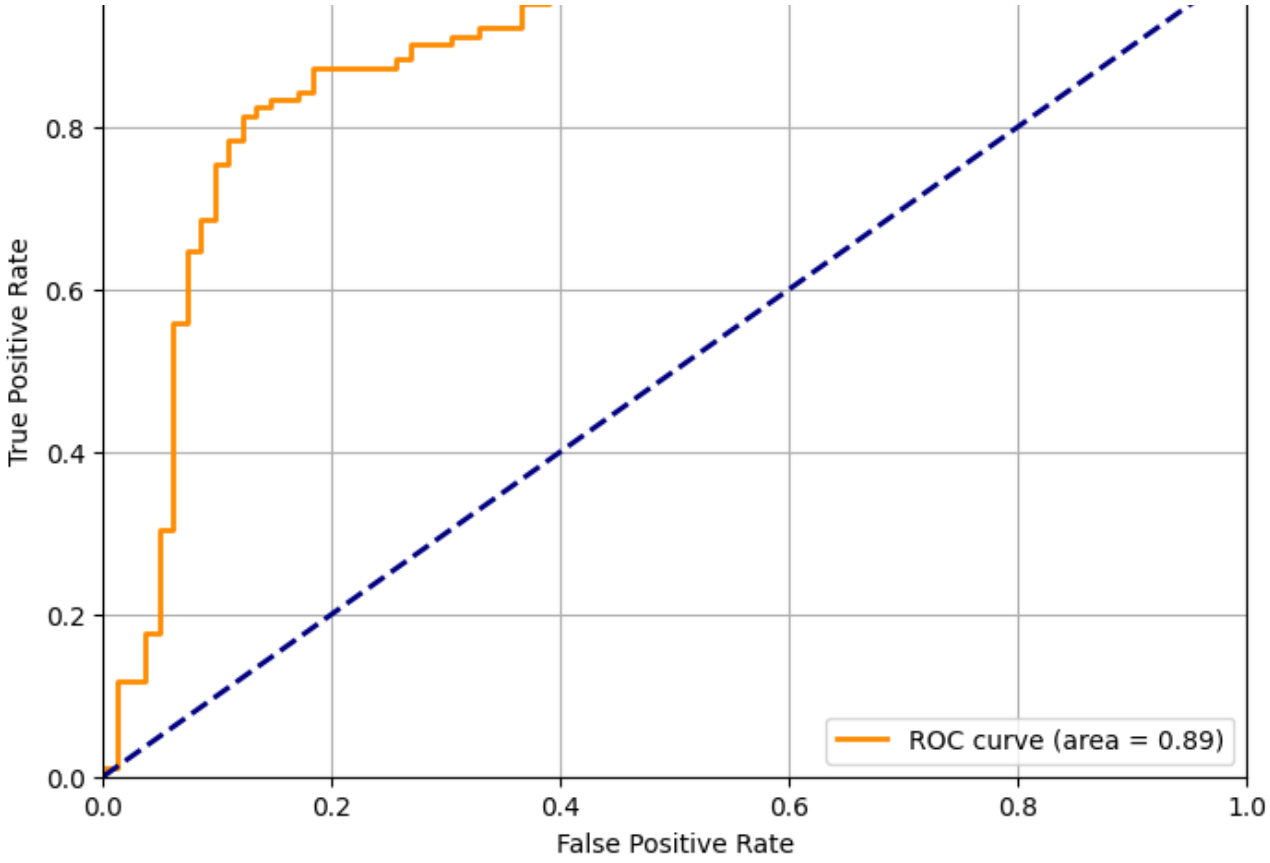
In [17]:

```
print("\n Final Evaluation Metrics (Regularized TabNet):")
for k, v in results.items():
    print(f"{k}: {v:.4f}")
```

```
 Final Evaluation Metrics (Regularized TabNet):
Accuracy: 0.8424
Precision: 0.8476
Recall: 0.8725
F1 Score: 0.8599
ROC AUC: 0.8920
```

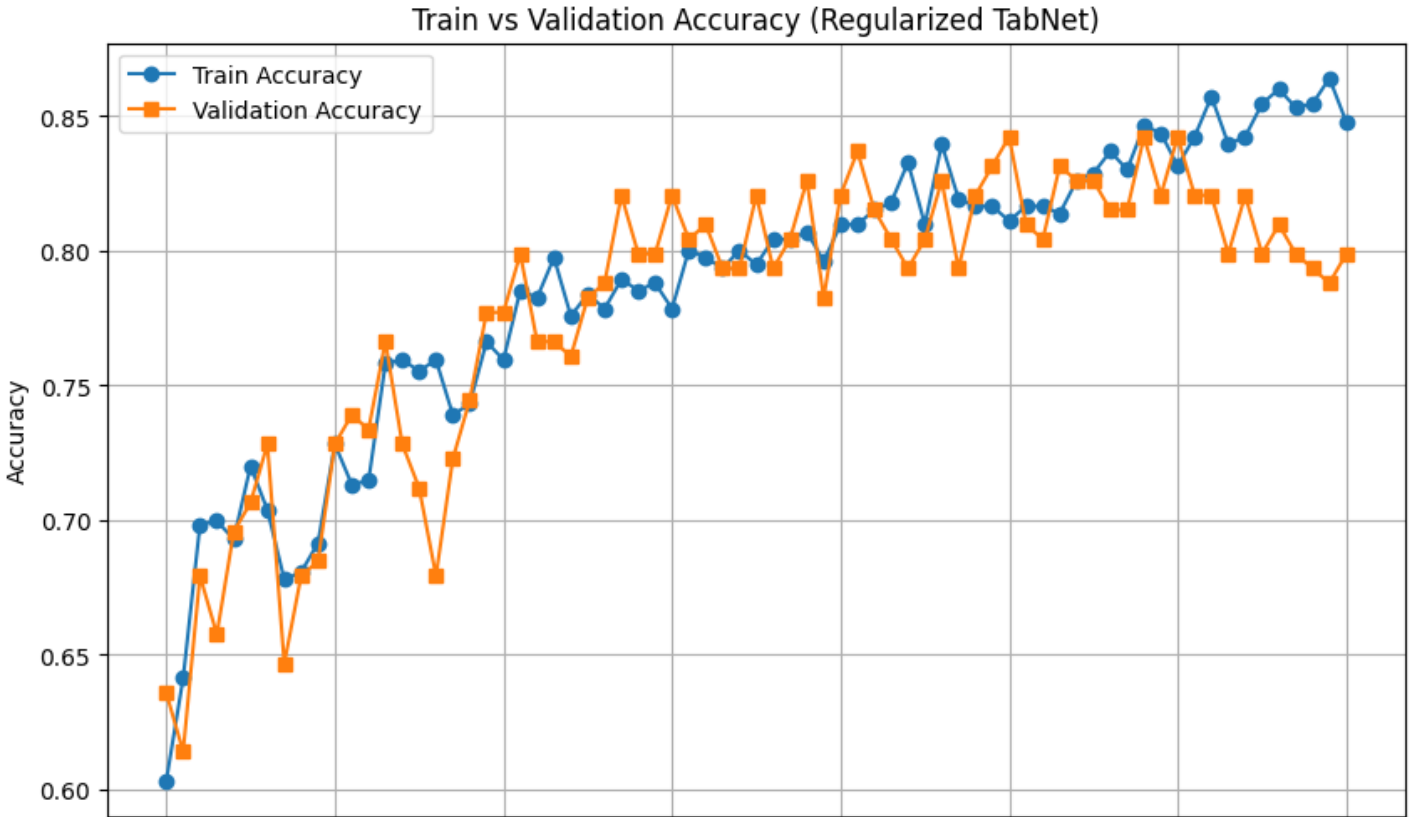
In [18]:

```
# ROC Curve
fpr, tpr, thresholds = roc_curve(y_test, y_pred_proba)
roc_auc = auc(fpr, tpr)

plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area = {roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic - TabNet')
plt.legend(loc="lower right")
plt.grid(True)
plt.show()
```



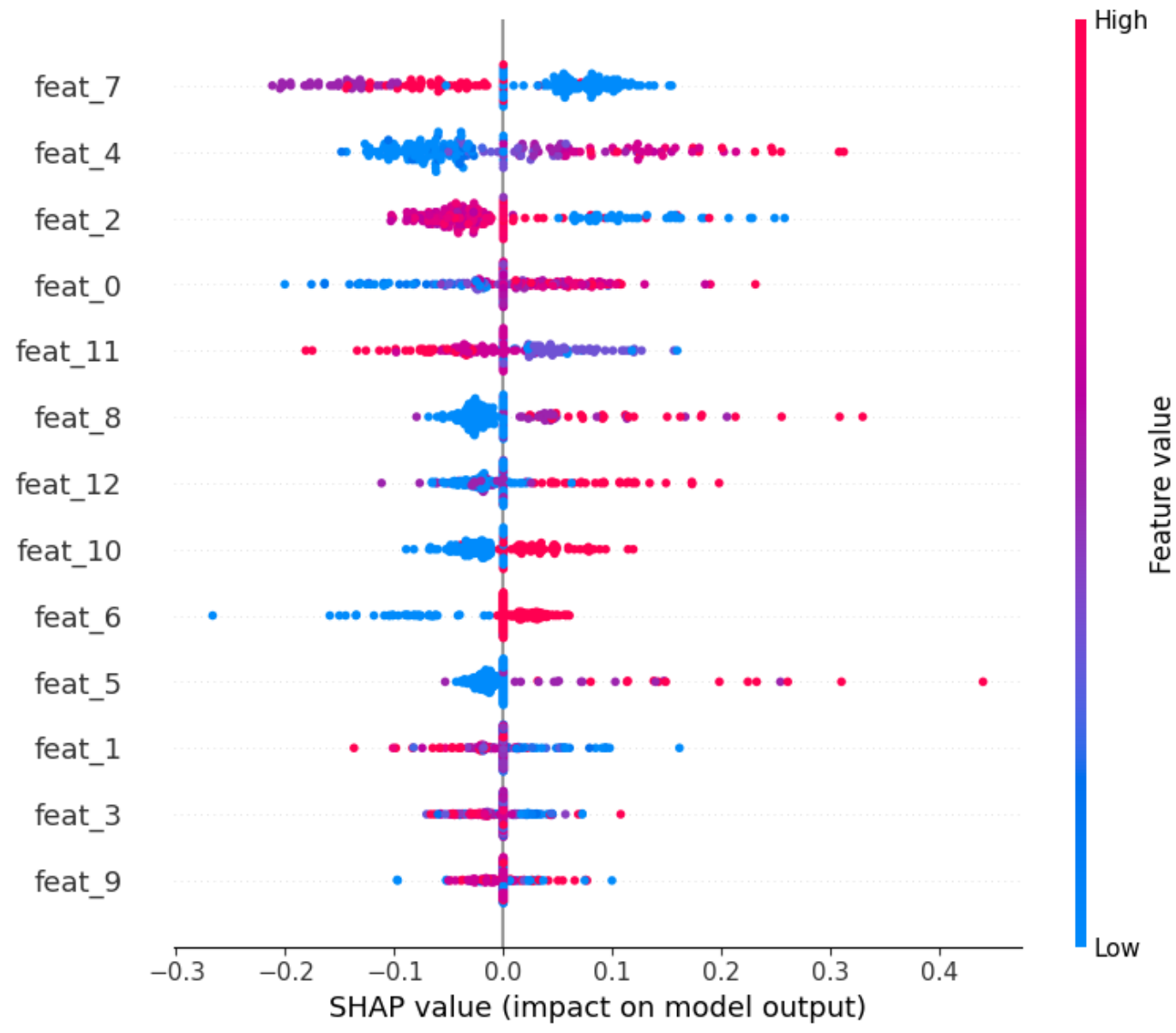
In [19]:

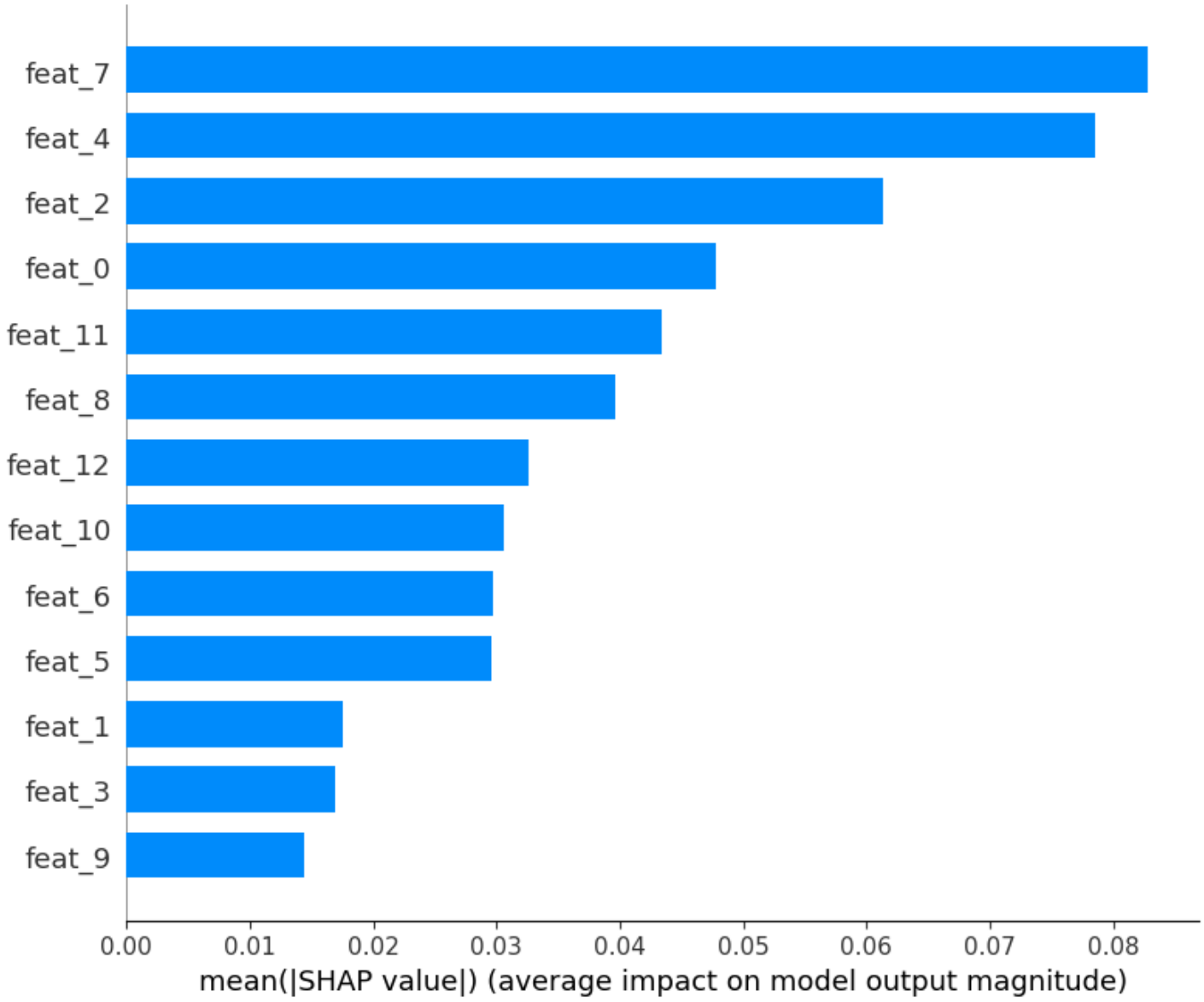
```
# Accuracy Plot
train_acc = tabnet.history['train_accuracy']
val_acc = tabnet.history['val_accuracy']

plt.figure(figsize=(10, 6))
plt.plot(train_acc, label='Train Accuracy', marker='o')
plt.plot(val_acc, label='Validation Accuracy', marker='s')
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.title("Train vs Validation Accuracy (Regularized TabNet)")
plt.legend()
plt.grid(True)
plt.show()
```



In [20]:

```
import shap
import pandas as pd
import numpy as np

# SHAP
X_sample = X_test[:200]

# Defining prediction function
predict_fn = lambda x: tabnet.predict_proba(x)[:, 1]
explainer = shap.KernelExplainer(predict_fn, X_train[:100])

# Computing SHAP values
shap_values = explainer.shap_values(X_sample)

# Determining feature names
if isinstance(X, pd.DataFrame):
    feature_names = X.columns
else:
    feature_names = [f'feat_{i}' for i in range(X.shape[1])]

X_sample_df = pd.DataFrame(X_sample, columns=feature_names)

# Summary plot
shap.summary_plot(shap_values, X_sample_df)
shap.summary_plot(shap_values, X_sample_df, plot_type='bar')
```




feat_7
feat_4
feat_2
feat_0
feat_11
feat_8
feat_12
feat_10
feat_6
feat_5
feat_1
feat_3
feat_9
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
mean(|SHAP value|) (average impact on model output magnitude)

In [20]: