In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
plt.rcParams['font.family'] = "Times New Roman"

import pycaret
import pycaret.classification as pycc
from pycaret.classification import *
import optuna

import imblearn
from imblearn.over_sampling import RandomOverSampler


%matplotlib inline
```

```
c:\Users\myleg\miniconda3\envs\cudapy_310\lib\site-packages\numpy\_distributor_init.py:30: UserWarning: loaded more than 1 DLL from .libs:
c:\Users\myleg\miniconda3\envs\cudapy_310\lib\site-packages\numpy\.libs\libopenblas.FB5AE2TYXYH2IJRDKGDGQ3XBKLKTF43H.gfortran-win_amd64.dll
c:\Users\myleg\miniconda3\envs\cudapy_310\lib\site-packages\numpy\.libs\libopenblas64__v0.3.21-gcc_10_3_0.dll
  warnings.warn("loaded more than 1 DLL from .libs:"
```

In [ ]:
```
print(pycaret.__version__)
print(imblearn.__version__)
```

```
3.1.0
0.11.0
```

Preparing data for classification machine learning models

In [ ]:
```
raw_df = pd.read_excel(r"C:\Users\myleg\OneDrive\바탕 화면\Thesis works\Long term outcome of bypass for ICAD\Long_term_result_Ischemic_disease_and_bypass_for_ML.xlsx")
raw_df
```

Out [ ]:

	age	sex	ipsilateral_postop_stroke	ipsilateral_symptomatic_postop_stroke	ipsilateral_asymptomatic_postop_stroke	any_postop_stroke	any_symptomatic_postop_stroke	any_asymptomatic_postop_stroke	any_postop_complication	any_symptomatic_postop_complication	...	infarct_cortical_internal_BDZ	infarct_large_deep	infarct_small_deep	infarct_old	infarct_no_infarct	collateral_Willisian	collateral_leptomeningeal	collateral_basal	collateral_transdural	poor_outcom
0	40	0		0	0	0	0	0	0	0	...	0	0	0	1	0	1	1	0	0	
1	49	1		0	0	0	1	1	0	1	...	0	0	0	0	0	1	1	0	1	
2	59	0		0	0	0	0	0	0	0	...	1	0	0	0	0	0	1	0	1	
3	59	0		0	0	0	0	0	0	0	...	1	0	0	0	0	0	1	0	0	
4	59	0		0	0	0	0	0	0	0	...	1	0	0	0	0	0	1	0	0	
...	...	...		...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
104	56	0		0	0	0	0	0	0	0	...	0	0	0	0	0	0	1	1	0	
105	52	1		0	0	0	0	0	0	0	...	0	0	0	0	0	1	1	1	0	
106	64	1		0	0	0	0	0	0	0	...	0	0	0	0	0	1	1	1	0	
107	67	0		0	0	0	1	1	0	1	...	1	0	0	0	0	1	1	0	1	
108	58	0		1	1	0	1	1	0	1	...	0	0	0	0	0	1	1	0	1	

109 rows × 81 columns

In [ ]:
```
# Defining training features

train_vars = ['age', 'BMI', 'first_event_to_last_event_days', 'first_event_to_treatment_days', 'last_event_to_treatment_days', 'n_intracranial_lesion_over_stenosis_50', 'initial_STAb_diameter', 'initial_STAs_diameter', 'initial_mRS', 'initial_mRS_grade', 'preop_mRS', 'preop_mRS_grade',  'single_ICAD', 'multiple_ICAD_bilateral', 'sex', 'previous_EVT_binary', 'previous_EVT', 'HTN', 'DM', 'DL', 'CAD'
        'smoking', 'prev_stroke', 'presentation', 'TIA', 'multiple_TIAs', 'infarction', 'multiple_infarction', 'hemorrhage','multiple_hemorrhage',
        'lesion_stenosis_occlusion', 'main_lesion_site', 'bypass_type', 'initial_bypass_patency', 'infarct_lobar', 'infarct_ICH', 'infarct_IVH', 'infarct_small_cortical',
        'infarct_cortical_BDZ', 'infarct_internal_BDZ', 'infarct_cortical_internal_BDZ', 'infarct_large_deep', 'infarct_small_deep', 'infarct_old',
        'infarct_no_infarct', 'collateral_Willisian', 'collateral_leptomeningeal', 'collateral_basal',
        'collateral_transdural', 'poor_outcome']

num_vars = ['age', 'BMI', 'first_event_to_last_event_days', 'first_event_to_treatment_days', 'last_event_to_treatment_days', 'n_intracranial_lesion_over_stenosis_50', 'initial_STAb_diameter', 'initial_STAs_diameter']

cat_vars = ['initial_mRS', 'initial_mRS_grade', 'preop_mRS', 'preop_mRS_grade',  'single_ICAD',
        'multiple_ICAD_bilateral', 'sex', 'previous_EVT_binary', 'previous_EVT', 'HTN', 'DM', 'DL', 'CAD', 'arrhythmia', 'FHx',
        'smoking', 'prev_stroke', 'presentation', 'TIA', 'multiple_TIAs',
        'infarction', 'multiple_infarction', 'hemorrhage',
        'multiple_hemorrhage', 'lesion_stenosis_occlusion', 'main_lesion_site', 'bypass_type',
        'initial_bypass_patency', 'infarct_lobar',
        'infarct_ICH', 'infarct_IVH', 'infarct_small_cortical',
        'infarct_cortical_BDZ', 'infarct_internal_BDZ',
        'infarct_cortical_internal_BDZ', 'infarct_large_deep',
        'infarct_small_deep', 'infarct_old', 'infarct_no_infarct', 'collateral_Willisian', 'collateral_leptomeningeal', 'collateral_basal',
        'collateral_transdural'}

ordered_cat_vars = {'initial_mRS', 'initial_mRS_grade', 'preop_mRS', 'preop_mRS_grade',  'single_ICAD',
        'multiple_ICAD_bilateral'}

non_ordered_cat_vars = {'sex', 'previous_EVT_binary', 'previous_EVT', 'HTN', 'DM', 'DL', 'CAD', 'arrhythmia', 'FHx',
        'smoking', 'prev_stroke', 'presentation', 'TIA', 'multiple_TIAs',
        'infarction', 'multiple_infarction', 'hemorrhage',
        'multiple_hemorrhage', 'lesion_stenosis_occlusion', 'main_lesion_site', 'bypass_type',
        'initial_bypass_patency', 'infarct_lobar',
        'infarct_ICH', 'infarct_IVH', 'infarct_small_cortical',
        'infarct_cortical_BDZ', 'infarct_internal_BDZ',
        'infarct_cortical_internal_BDZ', 'infarct_large_deep',
        'infarct_small_deep', 'infarct_old', 'infarct_no_infarct', 'collateral_Willisian', 'collateral_leptomeningeal', 'collateral_basal',
        'collateral_transdural'}
```

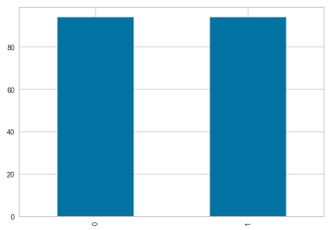
In [ ]:
```
dataset = raw_df[train_vars]
```

In [ ]:

In [ ]:
```
# Imbalanced data handling - Random Over-sampling : to overcome the limitations of imbalanced data
ros = RandomOverSampler(random_state=123)
X = dataset.drop(['poor_outcome'], axis=1)
y = dataset['poor_outcome']
X_resampled, y_resampled = ros.fit_resample(X, y)
```

In [ ]:
```
y_resampled.value_counts().plot(kind='bar')
```

Out [ ]:
```
<Axes: >
```



In [ ]:
```
# setting up classification models
ros_dataset = X_resampled
ros_dataset['poor_outcome'] = y_resampled

ros_clf = setup(data=ros_dataset, target='poor_outcome', categorical_features=cat_vars, numeric_features=num_vars, session_id=123, use_gpu=True)
```

In [ ]:
```
best_model = compare_models()
```

Initiated	09:37:03
Status	Selecting Estimator
Estimator	Logistic Regression

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
qda	Quadratic Discriminant Analysis	0.9923	0.9917	0.9833	1.0000	0.9909	0.9843	0.9854	0.1990
et	Extra Trees Classifier	0.9769	0.9929	0.9833	0.9750	0.9776	0.9530	0.9562	0.3860
rf	Random Forest Classifier	0.9615	0.9976	1.0000	0.9339	0.9646	0.9224	0.9276	0.3970
ada	Ada Boost Classifier	0.9165	0.9599	1.0000	0.8669	0.9253	0.8333	0.8508	0.2280
lightgbm	Light Gradient Boosting Machine	0.9159	0.9786	1.0000	0.8682	0.9261	0.8305	0.8486	0.4700
lr	Logistic Regression	0.8940	0.9214	1.0000	0.8410	0.9083	0.7859	0.8119	0.4570
dt	Decision Tree Classifier	0.8923	0.8905	1.0000	0.8345	0.9067	0.7817	0.8064	0.1900
gbc	Gradient Boosting Classifier	0.8846	0.9833	1.0000	0.8252	0.9003	0.7682	0.7958	0.2330
ridge	Ridge Classifier	0.8709	0.0000	1.0000	0.8164	0.8933	0.7398	0.7745	0.1900
lda	Linear Discriminant Analysis	0.7786	0.7488	1.0000	0.7026	0.8225	0.5530	0.6197	0.1970
nb	Naive Bayes	0.7335	0.7560	1.0000	0.6664	0.7947	0.4694	0.5504	0.1920
knn	K Neighbors Classifier	0.7330	0.8284	0.8929	0.7029	0.7775	0.4571	0.4983	0.2990
svm	SVM - Linear Kernel	0.4973	0.0000	0.3476	0.3528	0.3075	0.0291	0.0403	0.1890
dummy	Dummy Classifier	0.4654	0.5000	0.5000	0.2346	0.3193	0.0000	0.0000	0.1860

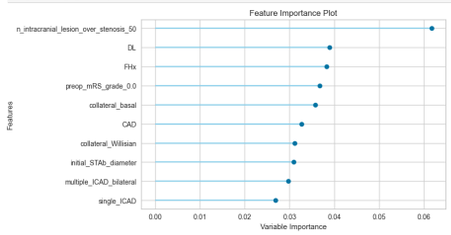
In [ ]:

In [ ]:

Evaluating important features for prediction

In [ ]:
```
# Important features from top models: Extra Trees Classifier
ros_et_model = ros_clf.create_model('et')
```

In [ ]:
```
plot_model(ros_et_model, plot='feature')
```



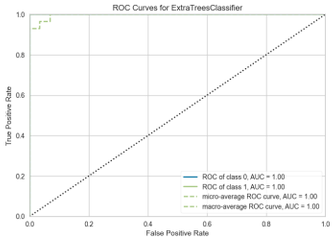
In [ ]:
```
tuned_ros_et_model = tune_model(ros_et_model, n_iter=100, optimize='f1')
```

Initiated	09:37:47
Status	Searching Hyperparameters
Estimator	Extra Trees Classifier

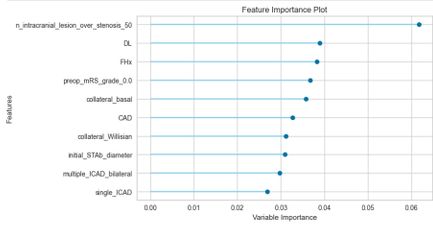
Fold	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.9286	1.0000	1.0000	0.8750	0.9333	0.8571	0.8660
1	0.9231	1.0000	1.0000	0.8750	0.9333	0.8434	0.8539
2	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
3	0.9231	1.0000	1.0000	0.8750	0.9333	0.8434	0.8539
4	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
5	0.8462	1.0000	1.0000	0.7778	0.8750	0.6829	0.7201
6	0.9231	1.0000	1.0000	0.8571	0.9231	0.8471	0.8571
7	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
8	0.9231	0.9762	0.8333	1.0000	0.9091	0.8434	0.8539
9	0.9231	1.0000	1.0000	0.8571	0.9231	0.8471	0.8571
Mean	0.9390	0.9976	0.9833	0.9117	0.9430	0.8764	0.8862
Std	0.0460	0.0071	0.0500	0.0769	0.0407	0.0941	0.0847

```
Fitting 10 folds for each of 100 candidates, totalling 1000 fits
Original model was better than the tuned model, hence it will be returned. NOTE: The display metrics are for the tuned model (not the original one).
```

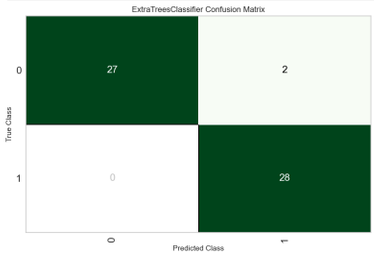
In [ ]:
```
# automatic fine tuning of the model
plot_model(tuned_ros_et_model, plot='auc')
```



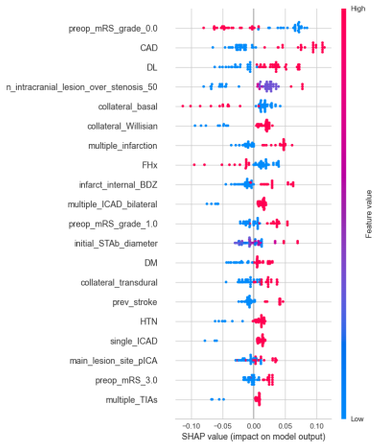
In [ ]:
```
plot_model(tuned_ros_et_model, plot='feature')
```



In [ ]:
```
plot_model(tuned_ros_et_model, plot='confusion_matrix')
```



In [ ]:
```
interpret_model(tuned_ros_et_model)
```



In [ ]:
```
fi_tuned_ros_et_model = pd.DataFrame({'Feature': tuned_ros_et_model.feature_names_in_, 'Value' : abs(tuned_ros_et_model.feature_importances_)}).sort_values(by='Value', ascending=False)
fi_tuned_ros_et_model
```

Out[ ]:

	Feature	Value
5	n_intracranial_lesion_over_stenosis_50	0.061800
34	DL	0.038907
37	FHx	0.038324
22	preop_mRS_grade_0.0	0.036714
78	collateral_basal	0.035807
...	...	...
62	initial_bypass_patency_5.0	0.000000
50	multiple_hemorrhage_0.0	0.000000
49	hemorrhage	0.000000
77	collateral_leptomeningeal_1.0	0.000000
44	presentation_3.0	0.000000

80 rows × 2 columns

In [ ]:

In [ ]:

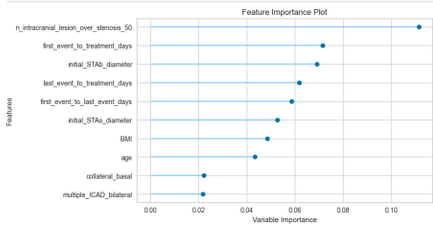
In [ ]:

In [ ]:
```
# Important features from top models: Random Forest Classifier
ros_rf_model = ros_clf.create_model('rf')
```

Initiated	09:40:02
Status	Fitting 10 Folds
Estimator	Random Forest Classifier

Fold	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	0.9231	1.0000	1.0000	0.8750	0.9333	0.8434	0.8539
2	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
3	0.9231	1.0000	1.0000	0.8750	0.9333	0.8434	0.8539
4	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
5	0.9231	1.0000	1.0000	0.8750	0.9333	0.8434	0.8539
6	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
7	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
8	0.9231	0.9762	1.0000	0.8571	0.9231	0.8471	0.8571
9	0.9231	1.0000	1.0000	0.8571	0.9231	0.8471	0.8571
Mean	0.9615	0.9976	1.0000	0.9339	0.9646	0.9224	0.9276
Std	0.0385	0.0071	0.0000	0.0664	0.0356	0.0776	0.0724

In [ ]:
```
plot_model(ros_rf_model, plot='feature')
```



In [ ]:
```
# automatic fine tuning of the model
tuned_ros_rf_model = tune_model(ros_rf_model, n_iter=100, optimize='f1')
```

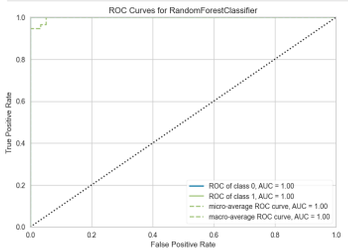
Initiated	09:40:07
Status	Searching Hyperparameters
Estimator	Random Forest Classifier

Fold	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	0.9231	1.0000	1.0000	0.8750	0.9333	0.8434	0.8539
2	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
3	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
4	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
5	0.9231	1.0000	1.0000	0.8750	0.9333	0.8434	0.8539
6	0.9231	1.0000	1.0000	0.8571	0.9231	0.8471	0.8571
7	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
8	0.9231	1.0000	0.8333	1.0000	0.9091	0.8434	0.8539

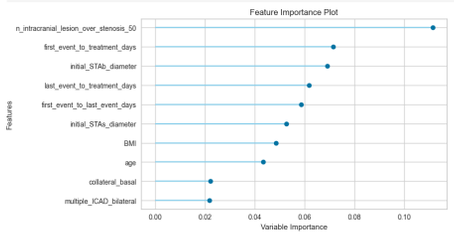
9	0.9231	1.0000	1.0000	0.8571	0.9231	0.8471	0.8571
Mean	0.9615	1.0000	0.9833	0.9464	0.9622	0.9224	0.9276
Std	0.0385	0.0000	0.0500	0.0659	0.0383	0.0776	0.0724

```
Fitting 10 folds for each of 100 candidates, totalling 1000 fits
Original model was better than the tuned model, hence it will be returned. NOTE: The display metrics are for the tuned model (not the original one).
```

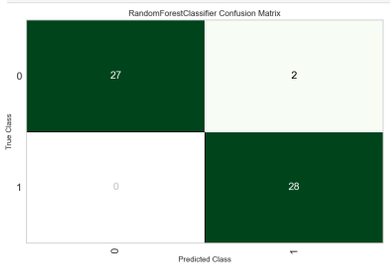
In [ ]:
```
plot_model(tuned_ros_rf_model, plot='auc')
```



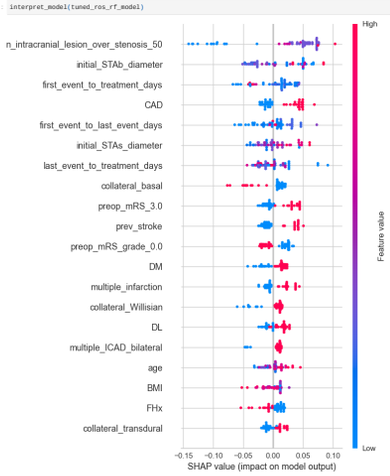
In [ ]:
```
plot_model(tuned_ros_rf_model, plot='feature')
```



In [ ]:
```
plot_model(tuned_ros_rf_model, plot='confusion_matrix')
```



In [ ]:
```
interpret_model(tuned_ros_rf_model)
```



In [ ]:
```
fi_tuned_ros_rf_model = pd.DataFrame({'Feature': tuned_ros_rf_model.feature_names_in_, 'Value' : abs(tuned_ros_rf_model.feature_importances_)}).sort_values(by='Value', ascending=False)
fi_tuned_ros_rf_model
```

Out[ ]:

	Feature	Value
5	n_intracranial_lesion_over_stenosis_50	0.111567
3	first_event_to_treatment_days	0.071425
6	initial_STAb_diameter	0.069088
4	last_event_to_treatment_days	0.061697
2	first_event_to_last_event_days	0.058715
...	...	...
44	presentation_3.0	0.000000
77	collateral_leptomeningeal_1.0	0.000000
58	bypass_type_4.0	0.000000
66	infarct_ICH_0.0	0.000000
50	multiple_hemorrhage_0.0	0.000000

80 rows × 2 columns

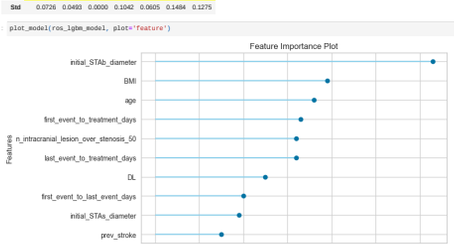
In [ ]:

In [ ]:
```
# Important features from top models: Light Gradient Boosting Machine
ros_lgbm_model = ros_clf.create_model('lightgbm')
```

Initiated	09:42:10
Status	Fitting 10 Folds
Estimator	Light Gradient Boosting Machine

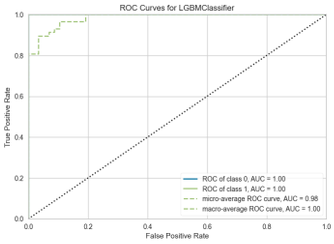
Fold	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.9286	1.0000	1.0000	0.8750	0.9333	0.8571	0.8660
1	0.9231	0.8333	1.0000	0.8750	0.9333	0.8434	0.8539
2	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
3	0.9231	1.0000	1.0000	0.8750	0.9333	0.8434	0.8539
4	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
5	0.7692	1.0000	1.0000	0.7000	0.8235	0.5185	0.5916
6	0.8462	0.9762	1.0000	0.7500	0.8571	0.6977	0.7319
7	0.8462	1.0000	1.0000	0.7500	0.8571	0.6977	0.7319
8	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
9	0.9231	0.9762	1.0000	0.8571	0.9231	0.8471	0.8571
Mean	0.9159	0.9786	1.0000	0.8682	0.9261	0.8305	0.8486
Std	0.0726	0.0493	0.0000	0.1042	0.0605	0.1484	0.1275

In [ ]:
```
plot_model(ros_lgbm_model, plot='feature')
```

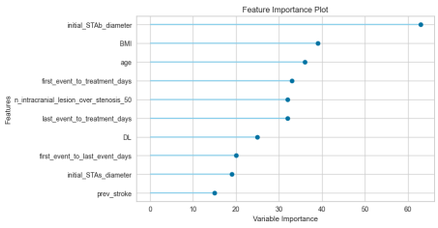


In [ ]:
```
# automatic fine tuning of the model
tuned_ros_lgbm_model = tune_model(ros_lgbm_model, n_iter=100, optimize='f1', tuner_verbose=-100)
```

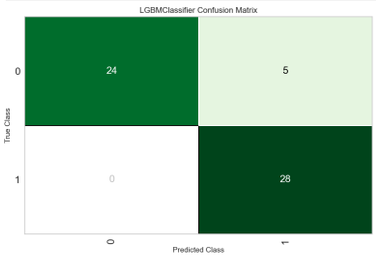
In [ ]:
```
plot_model(tuned_ros_lgbm_model, plot='auc')
```



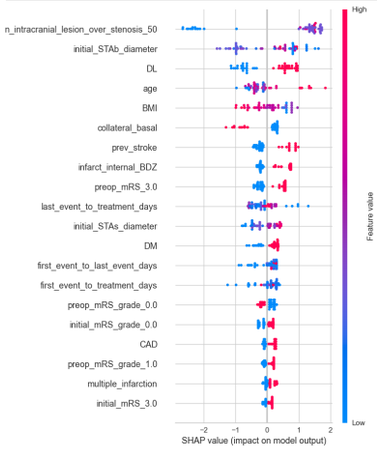
In [ ]:
```
plot_model(tuned_ros_lgbm_model, plot='feature')
```



In [ ]:
```
plot_model(tuned_ros_lgbm_model, plot='confusion_matrix')
```



In [ ]:
```
interpret_model(tuned_ros_lgbm_model)
```



In [ ]:
```
interpret_model(tuned_ros_lgbm_model, plot='pdp')
```

In [ ]:
```
fi_tuned_ros_lgbm_model = pd.DataFrame({'Feature': tuned_ros_lgbm_model.feature_name_, 'Value' : abs(tuned_ros_lgbm_model.feature_importances_)}).sort_values(by='Value', ascending=False)
fi_tuned_ros_lgbm_model
```

Out[ ]:

	Feature	Value
6	initial_STAb_diameter	63
1	BMI	39
0	age	36
3	first_event_to_treatment_days	33
4	last_event_to_treatment_days	32
...	...	...
46	multiple_TIAs	0
49	hemorrhage	0
50	multiple_hemorrhage_0.0	0
25	multiple_ICAD_bilateral	0
79	collateral_transdural	0

80 rows × 2 columns

In [ ]:

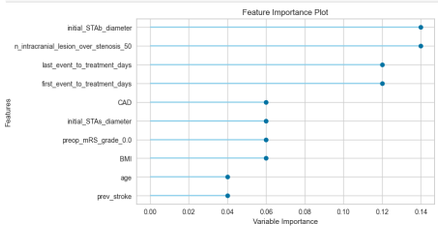
In [ ]:

In [ ]:
```
# Important features from top models: Ada Boost Classifier
ros_ada_model = ros_clf.create_model('ada')
```

Initiated	09:52:28
Status	Fitting 10 Folds
Estimator	Ada Boost Classifier

Fold	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.8571	0.9796	1.0000	0.7778	0.8750	0.7143	0.7454
1	0.8462	1.0000	1.0000	0.7778	0.8750	0.6829	0.7201
2	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
3	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
4	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
5	0.9231	0.9286	1.0000	0.8750	0.9333	0.8434	0.8539
6	0.7692	0.8571	1.0000	0.6667	0.8000	0.5517	0.6172
7	0.9231	1.0000	1.0000	0.8571	0.9231	0.8471	0.8571
8	0.9231	0.8571	1.0000	0.8571	0.9231	0.8471	0.8571
9	0.9231	0.9762	1.0000	0.8571	0.9231	0.8471	0.8571
Mean	0.9165	0.9599	1.0000	0.8669	0.9253	0.8333	0.8508
Std	0.0716	0.0555	0.0000	0.1046	0.0613	0.1415	0.1223

In [ ]:
```
plot_model(ros_ada_model, plot='feature')
```



In [ ]:
```
# automatic fine tuning of the model
tuned_ros_ada_model = tune_model(ros_ada_model, n_iter=100, optimize='f1')
```

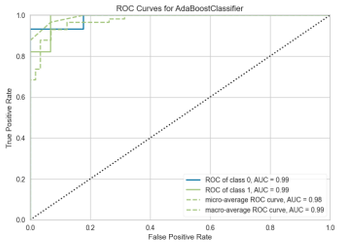
Initiated	09:52:31
Status	Loading Dependencies
Estimator	Ada Boost Classifier

Fold	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	0.9231	0.8810	1.0000	0.8750	0.9333	0.8434	0.8539
2	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
3	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
4	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
5	0.9231	1.0000	1.0000	0.8750	0.9333	0.8434	0.8539
6	0.8462	1.0000	1.0000	0.7500	0.8571	0.6977	0.7319
7	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

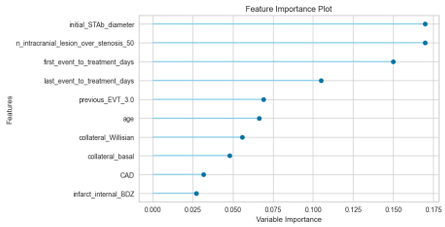
8	0.8462	0.8810	1.0000	0.7500	0.8571	0.6977	0.7319
9	0.9231	1.0000	1.0000	0.8571	0.9231	0.8471	0.8571
Mean	0.9462	0.9762	1.0000	0.9107	0.9504	0.8929	0.9029
Std	0.0601	0.0476	0.0000	0.0985	0.0557	0.1186	0.1061

```
Fitting 10 folds for each of 100 candidates, totalling 1000 fits
```

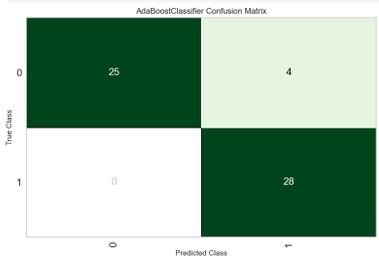
In [ ]:
```
plot_model(tuned_ros_ada_model, plot='auc')
```



In [ ]:
```
plot_model(tuned_ros_ada_model, plot='feature')
```



In [ ]:
```
plot_model(tuned_ros_ada_model, plot='confusion_matrix')
```



In [ ]:
```
fi_tuned_ros_ada_model = pd.DataFrame({'Feature': tuned_ros_ada_model.feature_names_in_, 'Value' : abs(tuned_ros_ada_model.feature_importances_)}).sort_values(by='Value', ascending=False)
fi_tuned_ros_ada_model
```

Out[ ]:

	Feature	Value
6	initial_STAb_diameter	0.169947
5	n_intracranial_lesion_over_stenosis_50	0.169855
3	first_event_to_treatment_days	0.150055
4	last_event_to_treatment_days	0.104950
29	previous_EVT_3.0	0.069018
...	...	...
39	smoking_2.0	0.000000
41	prev_stroke	0.000000
42	presentation_2.0	0.000000
43	presentation_3.0	0.000000
79	collateral_transdural	0.000000

80 rows × 2 columns

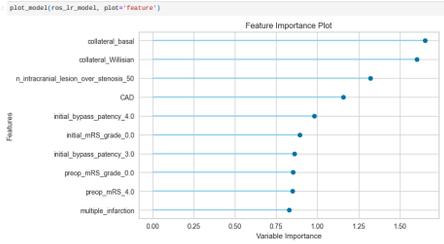
In [ ]:

In [ ]:
```
# Important features from top models: Logistic Regression
ros_lr_model = ros_clf.create_model('lr')
```

Initiated	09:53:14
Status	Initializing CV
Estimator	Logistic Regression

Fold	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.7857	0.8571	1.0000	0.7000	0.8235	0.5714	0.6325
1	0.8462	0.8333	1.0000	0.7778	0.8750	0.6829	0.7201
2	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
3	0.9231	0.9762	1.0000	0.8750	0.9333	0.8434	0.8539
4	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
5	0.7692	0.6667	1.0000	0.7000	0.8235	0.5185	0.5916
6	0.8462	1.0000	1.0000	0.7500	0.8571	0.6977	0.7319
7	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
8	0.8462	0.9048	1.0000	0.7500	0.8571	0.6977	0.7319
9	0.9231	0.9762	1.0000	0.8571	0.9231	0.8471	0.8571
Mean	0.8840	0.9214	1.0000	0.8410	0.9093	0.7859	0.8119
Std	0.0834	0.1038	0.0000	0.1173	0.0684	0.1698	0.1456

In [ ]:
```
plot_model(ros_lr_model, plot='feature')
```



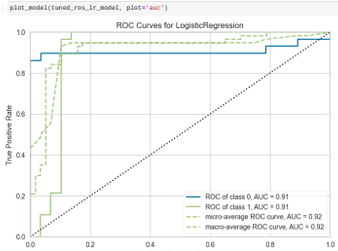
In [ ]:
```
# automatic fine tuning of the model
tuned_ros_lr_model = tune_model(ros_lr_model, n_iter=100, optimize='f1')
```

Initiated	09:53:19
Status	Searching Hyperparameters
Estimator	Logistic Regression

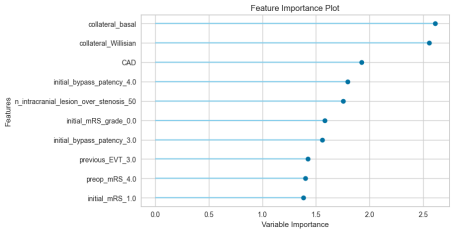
Fold	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.7857	0.8571	1.0000	0.7000	0.8235	0.5714	0.6325
1	0.7692	0.7857	1.0000	0.7000	0.8235	0.5185	0.5916
2	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
3	0.9231	1.0000	1.0000	0.8750	0.9333	0.8434	0.8539
4	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
5	0.8462	0.6905	1.0000	0.7778	0.8750	0.6829	0.7201
6	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
7	0.9231	1.0000	1.0000	0.8571	0.9231	0.8471	0.8571
8	0.9231	1.0000	1.0000	0.8571	0.9231	0.8471	0.8571
9	0.9231	0.8571	1.0000	0.8571	0.9231	0.8471	0.8571
Mean	0.9093	0.9190	1.0000	0.8624	0.9225	0.8157	0.8369
Std	0.0802	0.1082	0.0000	0.1081	0.0632	0.1647	0.1403

```
Fitting 10 folds for each of 100 candidates, totalling 1000 fits
```

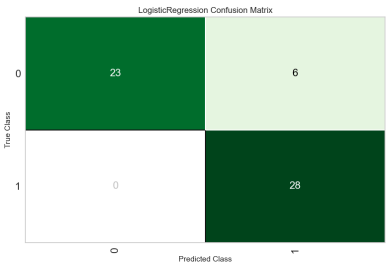
In [ ]:
```
plot_model(tuned_ros_lr_model, plot='auc')
```



In [ ]:
```
plot_model(tuned_ros_lr_model, plot='feature')
```



In [ ]:
```
plot_model(tuned_ros_lr_model, plot='confusion_matrix')
```



In [ ]:
```
fi_tuned_ros_lr_model = pd.DataFrame({'Feature': tuned_ros_lr_model.feature_names_in_, 'Value' : abs(tuned_ros_lr_model.coef_[0])}).sort_values(by='Value', ascending=False)
fi_tuned_ros_lr_model
```

Out[ ]:

	Feature	Value
78	collateral_basal	2.613370
76	collateral_Willisian	2.553530
35	CAD	1.922802
60	initial_bypass_patency_4.0	1.797376
5	n_intracranial_lesion_over_stenosis_50	1.752488
...	...	...
49	hemorrhage	0.022231
10	initial_mRS_2.0	0.017541
2	first_event_to_last_event_days	0.002660
3	first_event_to_treatment_days	0.001939
4	last_event_to_treatment_days	0.000721

80 rows × 2 columns

In [ ]:

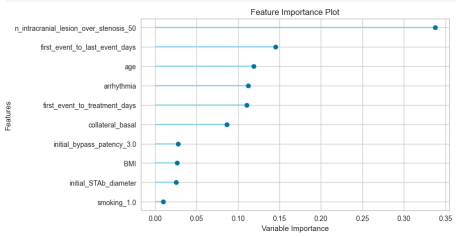
In [ ]:

In [ ]:
```
# Important features from top models: Decision Tree Classifier
ros_dt_model = ros_clf.create_model('dt')
```

	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
Fold							
0	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	0.9231	0.9167	1.0000	0.8750	0.9333	0.8434	0.8539
2	0.8462	0.8333	1.0000	0.7778	0.8750	0.6829	0.7201
3	0.7692	0.7500	1.0000	0.7000	0.8235	0.5185	0.5916
4	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
5	0.8462	0.8333	1.0000	0.7778	0.8750	0.6829	0.7201
6	0.8462	0.8571	1.0000	0.7500	0.8571	0.6977	0.7319
7	0.9231	0.9286	1.0000	0.8571	0.9231	0.8471	0.8571
8	0.8462	0.8571	1.0000	0.7500	0.8571	0.6977	0.7319
9	0.9231	0.9286	1.0000	0.8571	0.9231	0.8471	0.8571
Mean	0.8923	0.8905	1.0000	0.8345	0.9067	0.7817	0.8064
Std	0.0705	0.0749	0.0000	0.0981	0.0570	0.1451	0.1243

```
Processing:   0%|          | 0/4 [00:00<?, ?it/s]
```

In [ ]:
```
plot_model(ros_dt_model, plot='feature')
```



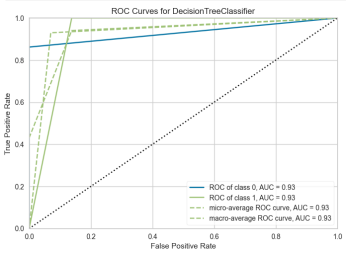
In [ ]:
```
# automatic fine tuning of the model
tuned_ros_dt_model = tune_model(ros_dt_model, n_iter=100, optimize='f1')
```

Initiated	09:53:52
Status	Loading Dependencies
Estimator	Decision Tree Classifier

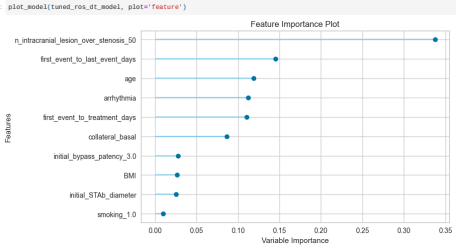
	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
Fold							
0	0.9286	1.0000	1.0000	0.8750	0.9333	0.8571	0.8660
1	0.8462	0.8333	1.0000	0.7778	0.8750	0.6829	0.7201
2	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
3	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
4	0.9231	0.9167	1.0000	0.8750	0.9333	0.8434	0.8539
5	0.8462	0.8214	1.0000	0.7778	0.8750	0.6829	0.7201
6	0.7692	0.7976	1.0000	0.6667	0.8000	0.5517	0.6172
7	0.9231	0.8929	1.0000	0.8571	0.9231	0.8471	0.8571
8	0.7692	0.7381	1.0000	0.6667	0.8000	0.5517	0.6172
9	0.9231	0.8929	1.0000	0.8571	0.9231	0.8471	0.8571
Mean	0.8929	0.8893	1.0000	0.8353	0.9063	0.7864	0.8109
Std	0.0787	0.0875	0.0000	0.1104	0.0664	0.1544	0.1312

```
Fitting 10 folds for each of 100 candidates, totalling 1000 fits
Original model was better than the tuned model, hence it will be returned. NOTE: The display metrics are for the tuned model (not the original one).
```

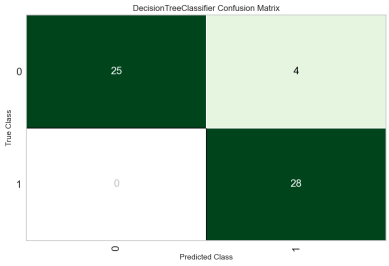
In [ ]:
```
plot_model(tuned_ros_dt_model, plot='auc')
```



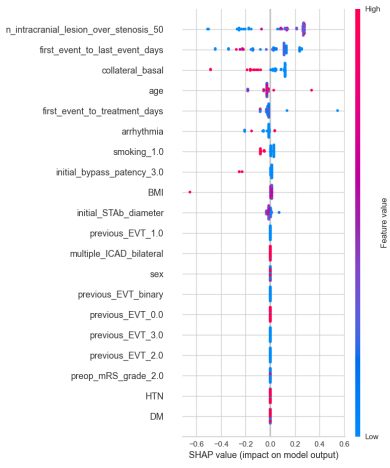
In [ ]:
```
plot_model(tuned_ros_dt_model, plot='feature')
```



In [ ]:
```
plot_model(tuned_ros_dt_model, plot='confusion_matrix')
```



In [ ]:
```
interpret_model(tuned_ros_dt_model)
```



In [ ]:
```
fi_tuned_ros_dt_model = pd.DataFrame({'Feature': tuned_ros_dt_model.feature_names_in_, 'Value' : abs(tuned_ros_dt_model.feature_importances_)}).sort_values(by='Value', ascending=False)
fi_tuned_ros_dt_model
```

Out[ ]:

	Feature	Value
5	n_intracranial_lesion_over_stenosis_50	0.337975
2	first_event_to_last_event_days	0.145059
0	age	0.118552
36	arrhythmia	0.112066
3	first_event_to_treatment_days	0.110638
...	...	...
28	previous_EVT_0.0	0.000000
27	previous_EVT_binary	0.000000
26	sex	0.000000
25	multiple_ICAD_bilateral	0.000000
79	collateral_transdural	0.000000

80 rows × 2 columns

In [ ]:

In [ ]:

Collecting and analysing important features from models

In [ ]:
```
print(fi_tuned_ros_et_model)
print(fi_tuned_ros_rf_model)
print(fi_tuned_ros_lgbm_model)
print(fi_tuned_ros_ada_model)
print(fi_tuned_ros_lr_model)
print(fi_tuned_ros_dt_model)
```

```
                                   Feature     Value
5   n_intracranial_lesion_over_stenosis_50  0.061808
34                                      DL  0.038907
37                                     FHx  0.038324
22                     preop_mRS_grade_0.0  0.036714
78                       collateral_basal   0.035807
..                                     ...       ...
62             initial_bypass_patency_5.0  0.000000
50                 multiple_hemorrhage_0.0  0.000000
49                                hemorrhage  0.000000
77           collateral_leptomeningeal_1.0  0.000000
44                       presentation_3.0  0.000000

[80 rows x 2 columns]
                                   Feature     Value
5   n_intracranial_lesion_over_stenosis_50  0.111567
3            first_event_to_treatment_days  0.071425
6                     initial_STAb_diameter  0.069088
4             last_event_to_treatment_days  0.061697
2           first_event_to_last_event_days  0.058715
..                                     ...       ...
44                       presentation_3.0  0.000000
77           collateral_leptomeningeal_1.0  0.000000
58                        bypass_type_4.0  0.000000
66                         infarct_ICH_0.0  0.000000
50                 multiple_hemorrhage_0.0  0.000000

[80 rows x 2 columns]
                          Feature  Value
6           initial_STAb_diameter     63
1                             BMI     39
0                             age     36
3   first_event_to_treatment_days     33
4    last_event_to_treatment_days     32
..                            ...    ...
46                   multiple_TIAs      0
49                       hemorrhage     0
50         multiple_hemorrhage_0.0      0
25         multiple_ICAD_bilateral      0
79           collateral_transdural      0

[80 rows x 2 columns]
                                   Feature     Value
6                     initial_STAb_diameter  0.169947
5   n_intracranial_lesion_over_stenosis_50  0.169855
3            first_event_to_treatment_days  0.150855
4             last_event_to_treatment_days  0.104050
29                     previous_EVT_3.0  0.069018
..                                     ...       ...
39                          smoking_2.0  0.000000
41                           prev_stroke  0.000000
42                       presentation_2.0  0.000000
43                       presentation_1.0  0.000000
79                  collateral_transdural  0.000000

[80 rows x 2 columns]
                                   Feature     Value
78                       collateral_basal  2.613370
76                   collateral_Willisian  2.553530
35                                    CAD  1.922802
60             initial_bypass_patency_4.0  1.797376
5   n_intracranial_lesion_over_stenosis_50  1.752488
..                                     ...       ...
49                                hemorrhage  0.022231
10                       initial_mRS_2.0  0.017541
2           first_event_to_last_event_days  0.002660
3            first_event_to_treatment_days  0.001939
4             last_event_to_treatment_days  0.000721

[80 rows x 2 columns]
                                   Feature     Value
5   n_intracranial_lesion_over_stenosis_50  0.337975
2           first_event_to_last_event_days  0.145059
0                                      age  0.118552
36                                arrhythmia  0.112066
3            first_event_to_treatment_days  0.110638
..                                     ...       ...
28                       previous_EVT_0.0  0.000000
27                     previous_EVT_binary  0.000000
26                                     sex  0.000000
25                 multiple_ICAD_bilateral  0.000000
79                   collateral_transdural  0.000000

[80 rows x 2 columns]
```

In [ ]:
```
fi_tuned_ros_et_model['Feature'].reset_index(drop=True)
```

Out[ ]:
```
0     n_intracranial_lesion_over_stenosis_50
1                                         DL
2                                        FHx
3                      preop_mRS_grade_0.0
4                        collateral_basal
                         ...
75             initial_bypass_patency_5.0
76                 multiple_hemorrhage_0.0
77                                hemorrhage
78           collateral_leptomeningeal_1.0
79                       presentation_3.0
Name: Feature, Length: 80, dtype: object
```

In [ ]:
```
fi_df = pd.concat([fi_tuned_ros_et_model['Feature'].reset_index(drop=True), fi_tuned_ros_rf_model['Feature'].reset_index(drop=True), fi_tuned_ros_lgbm_model['Feature'].reset_index(drop=True), fi_tuned_ros_ada_model['Feature'].reset_index(drop=True), fi_tuned_ros_lr_model['Feature'].reset_index(drop=True), fi_tuned_ros_dt_model['Feature'].reset_index(drop=True)], axis=1, ignore_index=True)
fi_df
```

Out[ ]:

	0	1	2	3	4	5
0	n_intracranial_lesion_over_stenosis_50	n_intracranial_lesion_over_stenosis_50	initial_STAb_diameter	initial_STAb_diameter	collateral_basal	n_intracranial_lesion_over_stenosis_50
1	DL	first_event_to_treatment_days	BMI	n_intracranial_lesion_over_stenosis_50	collateral_Willisian	first_event_to_last_event_days
2	FHx	initial_STAb_diameter	age	first_event_to_treatment_days	CAD	age
3	preop_mRS_grade_0.0	last_event_to_treatment_days	first_event_to_treatment_days	last_event_to_treatment_days	initial_bypass_patency_4.0	arrhythmia
4	collateral_basal	first_event_to_last_event_days	last_event_to_treatment_days	previous_EVT_3.0	n_intracranial_lesion_over_stenosis_50	first_event_to_treatment_days
...	...	...	...	...	...	...
75	initial_bypass_patency_5.0	presentation_3.0	multiple_TIAs	smoking_2.0	hemorrhage	previous_EVT_0.0
76	multiple_hemorrhage_0.0	collateral_leptomeningeal_1.0	hemorrhage	prev_stroke	initial_mRS_2.0	previous_EVT_binary
77	hemorrhage	bypass_type_4.0	multiple_hemorrhage_0.0	presentation_2.0	first_event_to_last_event_days	sex
78	collateral_leptomeningeal_1.0	infarct_ICH_0.0	multiple_ICAD_bilateral	presentation_1.0	first_event_to_treatment_days	multiple_ICAD_bilateral
79	presentation_3.0	multiple_hemorrhage_0.0	collateral_transdural	collateral_transdural	last_event_to_treatment_days	collateral_transdural

80 rows × 6 columns

In [ ]:
```
# Top 10 feature importance from models
top_10_fi_df = fi_df.head(10)
top_10_fi_df
```

Out[ ]:

	0	1	2	3	4	5
0	n_intracranial_lesion_over_stenosis_50	n_intracranial_lesion_over_stenosis_50	initial_STAb_diameter	initial_STAb_diameter	collateral_basal	n_intracranial_lesion_over_stenosis_50
1	DL	first_event_to_treatment_days	BMI	n_intracranial_lesion_over_stenosis_50	collateral_Willisian	first_event_to_last_event_days
2	FHx	initial_STAb_diameter	age	first_event_to_treatment_days	CAD	age
3	preop_mRS_grade_0.0	last_event_to_treatment_days	first_event_to_treatment_days	last_event_to_treatment_days	initial_bypass_patency_4.0	arrhythmia
4	collateral_basal	first_event_to_last_event_days	last_event_to_treatment_days	previous_EVT_3.0	n_intracranial_lesion_over_stenosis_50	first_event_to_treatment_days
5	CAD	initial_STAs_diameter	n_intracranial_lesion_over_stenosis_50	age	initial_mRS_grade_0.0	collateral_basal
6	collateral_Willisian	BMI	DL	collateral_Willisian	initial_bypass_patency_3.0	initial_bypass_patency_3.0
7	initial_STAb_diameter	age	first_event_to_last_event_days	collateral_basal	previous_EVT_3.0	BMI
8	multiple_ICAD_bilateral	collateral_basal	initial_STAs_diameter	CAD	preop_mRS_4.0	initial_STAb_diameter
9	single_ICAD	multiple_ICAD_bilateral	prev_stroke	infarct_internal_BDZ	initial_mRS_1.0	smoking_1.0

In [ ]:
```
fi_tuned_ros_et_model['Feature'].reset_index(drop=True)[:10]
```

Out[ ]:
```
0    n_intracranial_lesion_over_stenosis_50
1                                        DL
2                                       FHx
3                     preop_mRS_grade_0.0
4                       collateral_basal
5                                       CAD
6                  collateral_Willisian
7                    initial_STAb_diameter
8                 multiple_ICAD_bilateral
9                            single_ICAD
Name: Feature, dtype: object
```

In [ ]:

In [ ]:
```
# Check features that are repeatedly included in the top 10 important features in models
fi_df2 = pd.concat([fi_tuned_ros_et_model['Feature'].reset_index(drop=True)[:10], fi_tuned_ros_rf_model['Feature'].reset_index(drop=True)[:10], fi_tuned_ros_lgbm_model['Feature'].reset_index(drop=True)[:10], fi_tuned_ros_ada_model['Feature'].reset_index(drop=True)[:10], fi_tuned_ros_lr_model['Feature'].reset_index(drop=True)[:10], fi_tuned_ros_dt_model['Feature'].reset_index(drop=True)[:10], axis
fi_df2
```

Out[ ]:
```
0     n_intracranial_lesion_over_stenosis_50
1                                        DL
2                                       FHx
3                     preop_mRS_grade_0.0
4                       collateral_basal
                        ...
125                     previous_EVT_0.0
126                   previous_EVT_binary
127                                   sex
128               multiple_ICAD_bilateral
129                 collateral_transdural
Name: Feature, Length: 130, dtype: object
```

In [ ]:
```
fi_df2.value_counts(ascending=False)
```

Out[ ]:
```
n_intracranial_lesion_over_stenosis_50    6
collateral_basal                          5
initial_STAb_diameter                     5
age                                       4
CAD                                       4
                                         ..
infarct_large_deep                        1
infarct_cortical_internal_BDZ             1
infarct_cortical_BDZ                      1
infarct_small_cortical                    1
collateral_transdural                     1
Name: Feature, Length: 80, dtype: int64
```

In [ ]:
```
# Features that are included more than 3 times in the top 10 feature importance among 6 models
fi_df2.value_counts(ascending=False)[fi_df2.value_counts(ascending=False)>3]
```

Out[ ]:
```
n_intracranial_lesion_over_stenosis_50    6
collateral_basal                          5
initial_STAb_diameter                     5
age                                       4
CAD                                       4
collateral_Willisian                      4
```

```
first_event_to_treatment_days          4
last_event_to_treatment_days           4
Name: Feature, dtype: int64
```

In [ ]: