lab03

November 28, 2025

[ ]:

```
%pip install catboost optuna optuna-integration[catboost]
```

```
Requirement already satisfied: catboost in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (1.2.8)
Requirement already satisfied: optuna in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (4.6.0)
Requirement already satisfied: optuna-integration[catboost] in /home/optert/ML-
labs-Tert/.venv/lib/python3.11/site-packages (4.6.0)
Requirement already satisfied: graphviz in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from catboost) (0.21)
Requirement already satisfied: matplotlib in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from catboost) (3.10.7)
Requirement already satisfied: numpy<3.0,>=1.16.0 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from catboost) (1.26.4)
Requirement already satisfied: pandas>=0.24 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from catboost) (2.3.3)
Requirement already satisfied: scipy in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from catboost) (1.16.3)
Requirement already satisfied: plotly in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from catboost) (6.4.0)
Requirement already satisfied: six in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from catboost) (1.17.0)
Requirement already satisfied: alembic>=1.5.0 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from optuna) (1.17.1)
Requirement already satisfied: colorlog in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from optuna) (6.10.1)
Requirement already satisfied: packaging>=20.0 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from optuna) (25.0)
Requirement already satisfied: sqlalchemy>=1.4.2 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from optuna) (2.0.44)
Requirement already satisfied: tqdm in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from optuna) (4.67.1)
Requirement already satisfied: PyYAML in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from optuna) (6.0.3)
Requirement already satisfied: Mako in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from alembic>=1.5.0->optuna) (1.3.10)
Requirement already satisfied: typing-extensions>=4.12 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from alembic>=1.5.0->optuna) (4.15.0)
```

```
Requirement already satisfied: python-dateutil>=2.8.2 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from pandas>=0.24->catboost)
(2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from pandas>=0.24->catboost) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from pandas>=0.24->catboost) (2025.2)
Requirement already satisfied: greenlet>=1 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from sqlalchemy>=1.4.2->optuna) (3.0.3)
Requirement already satisfied: MarkupSafe>=0.9.2 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from Mako->alembic>=1.5.0->optuna)
(3.0.3)
Requirement already satisfied: contourpy>=1.0.1 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from matplotlib->catboost) (1.3.3)
Requirement already satisfied: cycler>=0.10 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from matplotlib->catboost) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from matplotlib->catboost) (4.60.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from matplotlib->catboost) (1.4.9)
Requirement already satisfied: pillow>=8 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from matplotlib->catboost) (12.0.0)
Requirement already satisfied: pyparsing>=3 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from matplotlib->catboost) (3.2.5)
Requirement already satisfied: narwhals>=1.15.1 in /home/optert/ML-labs-
Tert/.venv/lib/python3.11/site-packages (from plotly->catboost) (2.10.2)
Note: you may need to restart the kernel to use updated packages.
```

[68]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import optuna
from sklearn.decomposition import PCA
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, precision_score, recall_score,␣
  ↪f1_score, roc_auc_score, roc_curve
from sklearn.preprocessing import LabelEncoder, OrdinalEncoder, OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.impute import SimpleImputer
from sklearn.model_selection import StratifiedKFold
from catboost import CatBoostClassifier
from optuna.samplers import TPESampler
from sklearn.model_selection import cross_val_score, StratifiedKFold
from optuna.samplers import TPESampler
from optuna.pruners import MedianPruner
```

[ ]:

```
import os
from pathlib import Path

IN_COLAB = "COLAB_GPU" in os.environ or "google.colab" in str(get_ipython())

if not IN_COLAB:
    ROOT = Path.cwd()

    if ROOT.name == "lab02":
        LAB_ROOT = ROOT
    elif (ROOT / "lab02").exists():
        LAB_ROOT = ROOT / "lab02"
    elif ROOT.name == "notebooks":
        LAB_ROOT = ROOT.parent
    else:
        raise FileNotFoundError("         lab02             .")

    DATA_RAW = LAB_ROOT / "data" / "raw"
    DATA_RAW.mkdir(parents=True, exist_ok=True)

    csv_path = DATA_RAW / "telco.csv"

else:
    from urllib.request import urlretrieve

    url = "https://raw.githubusercontent.com/OlegTertychnyi/ML-labs-Tert/main/
 ↪lab03/data/raw/telco.csv"
    csv_path = Path("/content/telco.csv")

    if not csv_path.exists():
        urlretrieve(url, csv_path)
        print(" Dataset downloaded to Colab")

csv_path
```

[ ]:

```
PosixPath('/home/optert/ML-labs-Tert/lab03/data/raw/telco.csv')
```

[69]:

```
df = pd.read_csv(csv_path)
df
```

```
[69]:       customerID  gender  SeniorCitizen Partner Dependents  tenure  \
      0     7590-VHVEG  Female              0     Yes         No       1
      1     5575-GNVDE    Male              0      No         No      34
      2     3668-QPYBK    Male              0      No         No       2
      3     7795-CFOCW    Male              0      No         No      45
      4     9237-HQITU  Female              0      No         No       2
      …          …     …            …     …        …     …
      7038  6840-RESVB    Male              0     Yes        Yes      24
      7039  2234-XADUH  Female              0     Yes        Yes      72
      7040  4801-JZAZL  Female              0     Yes        Yes      11
      7041  8361-LTMKD    Male              1     Yes         No       4
      7042  3186-AJIEK    Male              0      No         No      66

           PhoneService     MultipleLines InternetService OnlineSecurity  …  \
      0              No  No phone service             DSL             No  …
      1             Yes                No             DSL            Yes  …
      2             Yes                No             DSL            Yes  …
      3              No  No phone service             DSL            Yes  …
      4             Yes                No     Fiber optic             No  …
      …           …               …             …            …  …
      7038          Yes               Yes             DSL            Yes  …
      7039          Yes               Yes     Fiber optic             No  …
      7040           No  No phone service             DSL            Yes  …
      7041          Yes               Yes     Fiber optic             No  …
      7042          Yes                No     Fiber optic            Yes  …

           DeviceProtection TechSupport StreamingTV StreamingMovies        Contract  \
      0                  No          No          No              No  Month-to-month
      1                 Yes          No          No              No        One year
      2                  No          No          No              No  Month-to-month
      3                 Yes         Yes          No              No        One year
      4                  No          No          No              No  Month-to-month
      …               …         …         …             …             …
      7038             Yes         Yes         Yes             Yes        One year
      7039             Yes          No         Yes             Yes        One year
      7040              No          No          No              No  Month-to-month
      7041              No          No          No              No  Month-to-month
      7042             Yes         Yes         Yes             Yes        Two year

           PaperlessBilling              PaymentMethod MonthlyCharges  TotalCharges  \
      0                 Yes           Electronic check          29.85         29.85
      1                  No               Mailed check          56.95        1889.5
      2                 Yes               Mailed check          53.85        108.15
      3                  No  Bank transfer (automatic)          42.30       1840.75
      4                 Yes           Electronic check          70.70        151.65
      …               …                        …            …           …
      7038             Yes               Mailed check          84.80        1990.5
```

```
7039                 Yes    Credit card (automatic)         103.20        7362.9
7040                 Yes           Electronic check          29.60        346.45
7041                 Yes               Mailed check          74.40         306.6
7042                 Yes  Bank transfer (automatic)         105.65        6844.5

     Churn
0       No
1       No
2      Yes
3       No
4      Yes
…     …
7038    No
7039    No
7040    No
7041   Yes
7042    No

[7043 rows x 21 columns]
```

[70]:

```
for col in df.select_dtypes(include=['object']).columns:
    df[col] = df[col].astype(str).str.strip()
df["TotalCharges"] = pd.to_numeric(df["TotalCharges"], errors="coerce")
df["Churn"] = (df["Churn"] == "Yes").astype(int)
df["TotalCharges"] = df["TotalCharges"].fillna(df["TotalCharges"].median())
```

[73]:

```
TARGET = "Churn"
ID_COLS = ["customerID"] if "customerID" in df.columns else []

X = df.drop(columns=[TARGET] + ID_COLS)
y = df[TARGET]

num_cols = X.select_dtypes(include=[np.number]).columns.tolist()
cat_cols = X.select_dtypes(include=["object"]).columns.tolist()

num_cols, cat_cols
```

[73]:

```
(['SeniorCitizen', 'tenure', 'MonthlyCharges', 'TotalCharges'],
 ['gender',
  'Partner',
  'Dependents',
  'PhoneService',
  'MultipleLines',
  'InternetService',
  'OnlineSecurity',
  'OnlineBackup',
  'DeviceProtection',
  'TechSupport',
  'StreamingTV',
  'StreamingMovies',
  'Contract',
  'PaperlessBilling',
  'PaymentMethod'])
```

[74]:

```
#
num_pipe = Pipeline([
    ("impute", SimpleImputer(strategy="median")),
    ("scale", StandardScaler())
])

#
cat_pipe = Pipeline([
    ("impute", SimpleImputer(strategy="most_frequent")),
    ("encode", OneHotEncoder(handle_unknown="ignore"))
])

#
preprocessor = ColumnTransformer([
    ("num", num_pipe, num_cols),
    ("cat", cat_pipe, cat_cols)
])

#
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)

#
models = {
    "Logistic Regression": LogisticRegression(max_iter=2000, random_state=42),
    "SVM (RBF)": SVC(kernel="rbf", probability=True, random_state=42),
    "KNN (k=7)": KNeighborsClassifier(n_neighbors=7),
    "Random Forest": RandomForestClassifier(n_estimators=500, random_state=42)
}
```

[75]:

```
#
pipelines = {
    model_name: Pipeline([
        ("prep", preprocessor),
        ("clf", model)
    ])
    for model_name, model in models.items()
}
```

```python
#
results = []

for model_name, pipe in pipelines.items():
    pipe.fit(X_train, y_train)

    prob = pipe.predict_proba(X_test)[:, 1]
    pred = pipe.predict(X_test)

    results.append({
        "Model": model_name,
        "Accuracy": accuracy_score(y_test, pred),
        "Precision": precision_score(y_test, pred),
        "Recall": recall_score(y_test, pred),
        "F1": f1_score(y_test, pred),
        "ROC-AUC": roc_auc_score(y_test, prob)
    })

results_df = pd.DataFrame(results)
results_df
```

[75]:

```
                 Model  Accuracy  Precision    Recall        F1   ROC-AUC
0  Logistic Regression  0.805536   0.657233  0.558824  0.604046  0.841874
1            SVM (RBF)  0.791341   0.641844  0.483957  0.551829  0.790491
2            KNN (k=7)  0.764372   0.554404  0.572193  0.563158  0.804810
3        Random Forest  0.776437   0.600683  0.470588  0.527736  0.819121
```

[77]:

```python
# CatBoost:            OneHot,

cat_features_idx = [X.columns.get_loc(c) for c in cat_cols]

cb_model = CatBoostClassifier(
    iterations=500,
    loss_function="Logloss",
    eval_metric="AUC",
    random_seed=42,
    verbose=False,
    cat_features=cat_features_idx
)

cb_model.fit(X_train, y_train)

y_pred = cb_model.predict(X_test)
y_proba = cb_model.predict_proba(X_test)[:, 1]

results.append({
    "Model": "CatBoost",
    "Accuracy": accuracy_score(y_test, y_pred),
    "Precision": precision_score(y_test, y_pred),
    "Recall": recall_score(y_test, y_pred),
    "F1": f1_score(y_test, y_pred),
    "ROC-AUC": roc_auc_score(y_test, y_proba)
})

pd.DataFrame(results)
```

[77]:

```
                 Model  Accuracy  Precision    Recall        F1   ROC-AUC
0  Logistic Regression  0.805536   0.657233  0.558824  0.604046  0.841874
1            SVM (RBF)  0.791341   0.641844  0.483957  0.551829  0.790491
2            KNN (k=7)  0.764372   0.554404  0.572193  0.563158  0.804810
3        Random Forest  0.776437   0.600683  0.470588  0.527736  0.819121
4             CatBoost  0.805536   0.676056  0.513369  0.583587  0.844270
```

[78]:

```
def build_logreg(C, penalty, l1_ratio=None):
    #           solver
    if penalty == "l1":
        solver = "liblinear"
        params = {"penalty": "l1", "C": C}
    elif penalty == "l2":
        solver = "lbfgs"
        params = {"penalty": "l2", "C": C}
    else:  # elasticnet
        solver = "saga"
        params = {"penalty": "elasticnet", "C": C, "l1_ratio": l1_ratio}

    model = LogisticRegression(
        solver=solver,
        max_iter=5000,
        random_state=42,
        **params
    )

    return Pipeline([
        ("prep", preprocessor),
        ("clf", model)
    ])


def objective(trial: optuna.Trial) -> float:
    penalty = trial.suggest_categorical("penalty", ["l1", "l2", "elasticnet"])
    C = trial.suggest_float("C", 1e-3, 1e2, log=True)

    l1_ratio = None
    if penalty == "elasticnet":
        l1_ratio = trial.suggest_float("l1_ratio", 0.0, 1.0)

    pipe = build_logreg(C=C, penalty=penalty, l1_ratio=l1_ratio)

    cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
    score = cross_val_score(pipe, X_train, y_train, cv=cv,
                            scoring="roc_auc", n_jobs=-1)

    return score.mean()


study_lr = optuna.create_study(direction="maximize",␣
 ↪sampler=TPESampler(seed=42))
study_lr.optimize(objective, n_trials=30, show_progress_bar=True)

print("Best AUC:", study_lr.best_value)
print("Best Params:", study_lr.best_params)
```

```
[I 2025-11-21 14:26:53,189] A new study created in memory with name: no-
name-13547912-ae20-4d86-89af-1b432b48a319

  0%|          | 0/30 [00:00<?, ?it/s]

[I 2025-11-21 14:26:59,848] Trial 0 finished with value: 0.8461231646197689 and
parameters: {'penalty': 'l2', 'C': 0.9846738873614566}. Best is trial 0 with
value: 0.8461231646197689.
[I 2025-11-21 14:27:03,789] Trial 1 finished with value: 0.846254108902029 and
parameters: {'penalty': 'l1', 'C': 21.42302175774105}. Best is trial 1 with
value: 0.846254108902029.
[I 2025-11-21 14:27:04,309] Trial 2 finished with value: 0.8462928493573113 and
parameters: {'penalty': 'l2', 'C': 70.72114131472235}. Best is trial 2 with
value: 0.8462928493573113.
[I 2025-11-21 14:27:04,756] Trial 3 finished with value: 0.8213951185326644 and
parameters: {'penalty': 'l1', 'C': 0.008260808399079604}. Best is trial 2 with
value: 0.8462928493573113.
[I 2025-11-21 14:27:05,209] Trial 4 finished with value: 0.8445429449751904 and
parameters: {'penalty': 'l2', 'C': 0.028585493941961918}. Best is trial 2 with
value: 0.8462928493573113.
[I 2025-11-21 14:27:05,751] Trial 5 finished with value: 0.8443335551539185 and
parameters: {'penalty': 'l1', 'C': 0.06789053271698488}. Best is trial 2 with
value: 0.8462928493573113.
[I 2025-11-21 14:27:06,281] Trial 6 finished with value: 0.8459348212817492 and
parameters: {'penalty': 'l2', 'C': 0.37253938395788866}. Best is trial 2 with
value: 0.8462928493573113.
[I 2025-11-21 14:27:07,094] Trial 7 finished with value: 0.8428136137978578 and
parameters: {'penalty': 'elasticnet', 'C': 0.007122305833333868, 'l1_ratio':
0.06505159298527952}. Best is trial 2 with value: 0.8462928493573113.
[I 2025-11-21 14:27:07,559] Trial 8 finished with value: 0.8446552990356977 and
```

```
parameters: {'penalty': 'l2', 'C': 0.03334792728637585}. Best is trial 2 with
value: 0.8462928493573113.
[I 2025-11-21 14:27:07,970] Trial 9 finished with value: 0.8420773194976992 and
parameters: {'penalty': 'l2', 'C': 0.00407559644007287}. Best is trial 2 with
value: 0.8462928493573113.
[I 2025-11-21 14:27:22,317] Trial 10 finished with value: 0.8462855826524814 and
parameters: {'penalty': 'elasticnet', 'C': 46.08678541950922, 'l1_ratio':
0.9695371107878658}. Best is trial 2 with value: 0.8462928493573113.
[I 2025-11-21 14:27:43,287] Trial 11 finished with value: 0.8462831727908314 and
parameters: {'penalty': 'elasticnet', 'C': 89.83132016522954, 'l1_ratio':
0.9076647952825176}. Best is trial 2 with value: 0.8462928493573113.
[I 2025-11-21 14:28:15,043] Trial 12 finished with value: 0.8462993678035176 and
parameters: {'penalty': 'elasticnet', 'C': 11.260271629505922, 'l1_ratio':
0.9702152277517139}. Best is trial 12 with value: 0.8462993678035176.
[I 2025-11-21 14:28:51,321] Trial 13 finished with value: 0.84628400798812 and
parameters: {'penalty': 'elasticnet', 'C': 6.2892520107372265, 'l1_ratio':
0.5585247307792902}. Best is trial 12 with value: 0.8462993678035176.
[I 2025-11-21 14:29:23,097] Trial 14 finished with value: 0.8462759305069749 and
parameters: {'penalty': 'elasticnet', 'C': 5.390363922402939, 'l1_ratio':
0.5780557316801694}. Best is trial 12 with value: 0.8462993678035176.
[I 2025-11-21 14:29:23,615] Trial 15 finished with value: 0.8462969266830216 and
parameters: {'penalty': 'l2', 'C': 6.831117665725749}. Best is trial 12 with
value: 0.8462993678035176.
[I 2025-11-21 14:29:39,632] Trial 16 finished with value: 0.8462355304023438 and
parameters: {'penalty': 'elasticnet', 'C': 2.3880566939107437, 'l1_ratio':
0.18318389605254987}. Best is trial 12 with value: 0.8462993678035176.
[I 2025-11-21 14:30:27,759] Trial 17 finished with value: 0.8463058247088714 and
parameters: {'penalty': 'elasticnet', 'C': 10.172075441673128, 'l1_ratio':
0.7593033273862684}. Best is trial 17 with value: 0.8463058247088714.
[I 2025-11-21 14:30:37,784] Trial 18 finished with value: 0.8460713882489129 and
parameters: {'penalty': 'elasticnet', 'C': 0.49899998666091444, 'l1_ratio':
0.7752033459935036}. Best is trial 17 with value: 0.8463058247088714.
[I 2025-11-21 14:31:54,699] Trial 19 finished with value: 0.8462718795559157 and
parameters: {'penalty': 'elasticnet', 'C': 17.962491814149992, 'l1_ratio':
0.7395095724659485}. Best is trial 17 with value: 0.8463058247088714.
[I 2025-11-21 14:32:11,983] Trial 20 finished with value: 0.8460949144377986 and
parameters: {'penalty': 'elasticnet', 'C': 1.7392244271251245, 'l1_ratio':
0.7613756157041719}. Best is trial 17 with value: 0.8463058247088714.
[I 2025-11-21 14:32:34,249] Trial 21 finished with value: 0.8462775501059274 and
parameters: {'penalty': 'elasticnet', 'C': 8.469553977729813, 'l1_ratio':
0.9936291448929787}. Best is trial 17 with value: 0.8463058247088714.
[I 2025-11-21 14:32:34,660] Trial 22 finished with value: 0.8393614728199157 and
parameters: {'penalty': 'l2', 'C': 0.0011633263858100054}. Best is trial 17 with
value: 0.8463058247088714.
[I 2025-11-21 14:32:38,661] Trial 23 finished with value: 0.8462880540549845 and
parameters: {'penalty': 'l1', 'C': 19.540027649806603}. Best is trial 17 with
value: 0.8463058247088714.
[I 2025-11-21 14:32:59,269] Trial 24 finished with value: 0.846292113797594 and
```

```
parameters: {'penalty': 'elasticnet', 'C': 3.125059166612226, 'l1_ratio':
0.35254572768281606}. Best is trial 17 with value: 0.8463058247088714.
[I 2025-11-21 14:32:59,771] Trial 25 finished with value: 0.8454498441951669 and
parameters: {'penalty': 'l2', 'C': 0.09997781281293299}. Best is trial 17 with
value: 0.8463058247088714.
[I 2025-11-21 14:33:49,327] Trial 26 finished with value: 0.8462985667955918 and
parameters: {'penalty': 'elasticnet', 'C': 13.457197497444264, 'l1_ratio':
0.8448345378628919}. Best is trial 17 with value: 0.8463058247088714.
[I 2025-11-21 14:34:03,045] Trial 27 finished with value: 0.846070631198739 and
parameters: {'penalty': 'elasticnet', 'C': 0.891474572640944, 'l1_ratio':
0.8296001148801442}. Best is trial 17 with value: 0.8463058247088714.
[I 2025-11-21 14:34:36,533] Trial 28 finished with value: 0.846287218857696 and
parameters: {'penalty': 'elasticnet', 'C': 28.753580717122208, 'l1_ratio':
0.6496777263540325}. Best is trial 17 with value: 0.8463058247088714.
[I 2025-11-21 14:34:51,862] Trial 29 finished with value: 0.8460690184376591 and
parameters: {'penalty': 'elasticnet', 'C': 0.9199988595892168, 'l1_ratio':
0.8493385231889122}. Best is trial 17 with value: 0.8463058247088714.
Best AUC: 0.8463058247088714
Best Params: {'penalty': 'elasticnet', 'C': 10.172075441673128, 'l1_ratio':
0.7593033273862684}
```

[79]:

```
#                            Optuna
best_lr_pipe = build_logreg(
    C=study_lr.best_params["C"],
    penalty=study_lr.best_params["penalty"],
    l1_ratio=study_lr.best_params.get("l1_ratio")
)

best_lr_pipe.fit(X_train, y_train)

#
lr_pred = best_lr_pipe.predict(X_test)
lr_proba = best_lr_pipe.predict_proba(X_test)[:, 1]

#
results.append({
    "Model": "LogReg (tuned)",
    "Accuracy": accuracy_score(y_test, lr_pred),
    "Precision": precision_score(y_test, lr_pred),
    "Recall": recall_score(y_test, lr_pred),
    "F1": f1_score(y_test, lr_pred),
    "ROC-AUC": roc_auc_score(y_test, lr_proba)
})
```

[80]:

```
pd.DataFrame(results)
```

[80]:

```
                 Model  Accuracy  Precision    Recall        F1   ROC-AUC
0  Logistic Regression  0.805536   0.657233  0.558824  0.604046  0.841874
1            SVM (RBF)  0.791341   0.641844  0.483957  0.551829  0.790491
2            KNN (k=7)  0.764372   0.554404  0.572193  0.563158  0.804810
3        Random Forest  0.776437   0.600683  0.470588  0.527736  0.819121
4             CatBoost  0.805536   0.676056  0.513369  0.583587  0.844270
5       LogReg (tuned)  0.804826   0.655172  0.558824  0.603175  0.841169
```

[85]:

```
#           catboost
cb_preprocessor = ColumnTransformer(
    transformers=[
        ('num', 'passthrough', num_cols),
        ('cat', 'passthrough', cat_cols),
    ],
    remainder='drop'
)

#                       ColumnTransformer(passthrough)
cat_idx = list(range(len(num_cols), len(num_cols) + len(cat_cols)))



def objective_cb(trial: optuna.Trial) -> float:

    params = {
      "learning_rate": trial.suggest_float("learning_rate", 0.03, 0.2,␣
 ↪log=True),
      "depth": trial.suggest_int("depth", 4, 6),
      "l2_leaf_reg": trial.suggest_float("l2_leaf_reg", 2.0, 8.0),
      "n_estimators": trial.suggest_int("n_estimators", 300, 1200),
      "bagging_temperature": trial.suggest_float("bagging_temperature", 0.0, 1.
 ↪0),
      "random_strength": trial.suggest_float("random_strength", 0.2, 1.5),
    }

    cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
    oof_scores = []

    for train_idx, valid_idx in cv.split(X_train, y_train):

        X_tr, X_va = X_train.iloc[train_idx], X_train.iloc[valid_idx]
        y_tr, y_va = y_train.iloc[train_idx], y_train.iloc[valid_idx]

        #
        pre = cb_preprocessor
        pre.fit(X_tr)
        X_tr_t = pre.transform(X_tr)
        X_va_t = pre.transform(X_va)

        model = CatBoostClassifier(
            loss_function="Logloss",
            eval_metric="AUC",
            random_seed=42,
            verbose=False,
            cat_features=cat_idx,
            **params
        )

        model.fit(
            X_tr_t, y_tr,
            eval_set=(X_va_t, y_va),
            early_stopping_rounds=100
        )

        y_va_proba = model.predict_proba(X_va_t)[:, 1]
        oof_scores.append(roc_auc_score(y_va, y_va_proba))

        trial.report(float(np.mean(oof_scores)), step=len(oof_scores))
        if trial.should_prune():
            raise optuna.TrialPruned()

    return float(np.mean(oof_scores))


study_cb = optuna.create_study(
    direction="maximize",
    sampler=TPESampler(seed=42),
    pruner=MedianPruner(n_startup_trials=10)
)

study_cb.optimize(objective_cb, n_trials=25, show_progress_bar=True)

print("Best CB AUC (CV):", study_cb.best_value)
print("Best CB params:", study_cb.best_params)
```

```
[I 2025-11-21 15:04:34,746] A new study created in memory with name: no-
name-72f32a14-ad10-4a04-9e54-2b5195afae05

  0%|          | 0/25 [00:00<?, ?it/s]

[I 2025-11-21 15:04:58,578] Trial 0 finished with value: 0.8490689059448876 and
parameters: {'learning_rate': 0.061053157936408836, 'depth': 6, 'l2_leaf_reg':
6.391963650868431, 'n_estimators': 839, 'bagging_temperature':
0.15601864044243652, 'random_strength': 0.40279287643706346}. Best is trial 0
with value: 0.8490689059448876.
```

```
[I 2025-11-21 15:05:32,586] Trial 1 finished with value: 0.8500720770375819 and
parameters: {'learning_rate': 0.033494758473134406, 'depth': 6, 'l2_leaf_reg':
5.606690070459253, 'n_estimators': 937, 'bagging_temperature':
0.020584494295802447, 'random_strength': 1.4608828078105927}. Best is trial 1
with value: 0.8500720770375819.
[I 2025-11-21 15:05:44,840] Trial 2 finished with value: 0.8500004473922319 and
parameters: {'learning_rate': 0.14553854298843344, 'depth': 4, 'l2_leaf_reg':
3.0909498032426037, 'n_estimators': 465, 'bagging_temperature':
0.3042422429595377, 'random_strength': 0.8821833611219092}. Best is trial 1 with
value: 0.8500720770375819.
[I 2025-11-21 15:06:04,039] Trial 3 finished with value: 0.8504845760941999 and
parameters: {'learning_rate': 0.06807764614983679, 'depth': 4, 'l2_leaf_reg':
5.671117368334277, 'n_estimators': 425, 'bagging_temperature':
0.29214464853521815, 'random_strength': 0.6762703962817993}. Best is trial 3
with value: 0.8504845760941999.
[I 2025-11-21 15:06:24,939] Trial 4 finished with value: 0.8490973075368927 and
parameters: {'learning_rate': 0.07126582239713602, 'depth': 6, 'l2_leaf_reg':
3.1980426929501586, 'n_estimators': 763, 'bagging_temperature':
0.5924145688620425, 'random_strength': 0.26038553653599705}. Best is trial 3
with value: 0.8504845760941999.
[I 2025-11-21 15:06:39,523] Trial 5 finished with value: 0.8512647006689061 and
parameters: {'learning_rate': 0.09499106760020673, 'depth': 4, 'l2_leaf_reg':
2.390309557911677, 'n_estimators': 1154, 'bagging_temperature':
0.9656320330745594, 'random_strength': 1.2509165525513994}. Best is trial 5 with
value: 0.8512647006689061.
[I 2025-11-21 15:07:01,412] Trial 6 finished with value: 0.8502828105008076 and
parameters: {'learning_rate': 0.053468155416899, 'depth': 4, 'l2_leaf_reg':
6.105398159072942, 'n_estimators': 696, 'bagging_temperature':
0.12203823484477883, 'random_strength': 0.8437299831446512}. Best is trial 5
with value: 0.8512647006689061.
[I 2025-11-21 15:07:34,153] Trial 7 finished with value: 0.8498331559100777 and
parameters: {'learning_rate': 0.032022428000607145, 'depth': 6, 'l2_leaf_reg':
3.5526798896001015, 'n_estimators': 896, 'bagging_temperature':
0.31171107608941095, 'random_strength': 0.876088427531154}. Best is trial 5 with
value: 0.8512647006689061.
[I 2025-11-21 15:07:51,579] Trial 8 finished with value: 0.8510134659982196 and
parameters: {'learning_rate': 0.08463707239000723, 'depth': 4, 'l2_leaf_reg':
7.817507766587351, 'n_estimators': 998, 'bagging_temperature':
0.9394989415641891, 'random_strength': 1.3632755555559435}. Best is trial 5 with
value: 0.8512647006689061.
[I 2025-11-21 15:08:08,957] Trial 9 finished with value: 0.8484031447883756 and
parameters: {'learning_rate': 0.09326877515737107, 'depth': 6, 'l2_leaf_reg':
2.5309550123115168, 'n_estimators': 476, 'bagging_temperature':
0.045227288910538066, 'random_strength': 0.6229294299922437}. Best is trial 5
with value: 0.8512647006689061.
[I 2025-11-21 15:08:18,694] Trial 10 pruned.
[I 2025-11-21 15:08:29,158] Trial 11 pruned.
[I 2025-11-21 15:08:46,464] Trial 12 finished with value: 0.851103143719714 and
```

```
parameters: {'learning_rate': 0.11250097158764306, 'depth': 4, 'l2_leaf_reg':
7.747857049445863, 'n_estimators': 1055, 'bagging_temperature':
0.7617805327747955, 'random_strength': 1.1856300139875224}. Best is trial 5 with
value: 0.8512647006689061.
[I 2025-11-21 15:09:01,821] Trial 13 finished with value: 0.8505059288164574 and
parameters: {'learning_rate': 0.1259013427444294, 'depth': 5, 'l2_leaf_reg':
4.437934368576814, 'n_estimators': 1072, 'bagging_temperature':
0.760596035264475, 'random_strength': 1.1089110643063502}. Best is trial 5 with
value: 0.8512647006689061.
[I 2025-11-21 15:09:13,874] Trial 14 finished with value: 0.8509299785050499 and
parameters: {'learning_rate': 0.1670917479157165, 'depth': 4, 'l2_leaf_reg':
2.1190426987904676, 'n_estimators': 1086, 'bagging_temperature':
0.7371824291629284, 'random_strength': 1.077988831634637}. Best is trial 5 with
value: 0.8512647006689061.
[I 2025-11-21 15:09:28,855] Trial 15 finished with value: 0.8516166308274229 and
parameters: {'learning_rate': 0.0968412457373984, 'depth': 4, 'l2_leaf_reg':
6.888702314029387, 'n_estimators': 618, 'bagging_temperature':
0.8095789183531144, 'random_strength': 1.4964672476949448}. Best is trial 15
with value: 0.8516166308274229.
[I 2025-11-21 15:09:52,598] Trial 16 finished with value: 0.8506173260632292 and
parameters: {'learning_rate': 0.05244335214874761, 'depth': 5, 'l2_leaf_reg':
6.8483962066829855, 'n_estimators': 625, 'bagging_temperature':
0.8467638526830317, 'random_strength': 1.4625324456243842}. Best is trial 15
with value: 0.8516166308274229.
[I 2025-11-21 15:10:08,168] Trial 17 finished with value: 0.8516926240122913 and
parameters: {'learning_rate': 0.0934604919339652, 'depth': 4, 'l2_leaf_reg':
4.84634432283171, 'n_estimators': 646, 'bagging_temperature':
0.5795660494660647, 'random_strength': 1.3335184557179816}. Best is trial 17
with value: 0.8516926240122913.
[I 2025-11-21 15:10:11,012] Trial 18 pruned.
[I 2025-11-21 15:10:15,029] Trial 19 pruned.
[I 2025-11-21 15:10:32,108] Trial 20 finished with value: 0.8512299252028568 and
parameters: {'learning_rate': 0.08084925472114562, 'depth': 4, 'l2_leaf_reg':
5.126041399127765, 'n_estimators': 586, 'bagging_temperature':
0.4578028138312887, 'random_strength': 1.4113618661911014}. Best is trial 17
with value: 0.8516926240122913.
[I 2025-11-21 15:10:34,746] Trial 21 pruned.
[I 2025-11-21 15:10:45,078] Trial 22 pruned.
[I 2025-11-21 15:10:47,098] Trial 23 pruned.
[I 2025-11-21 15:10:49,773] Trial 24 pruned.
Best CB AUC (CV): 0.8516926240122913
Best CB params: {'learning_rate': 0.0934604919339652, 'depth': 4, 'l2_leaf_reg':
4.84634432283171, 'n_estimators': 646, 'bagging_temperature':
0.5795660494660647, 'random_strength': 1.3335184557179816}
```

[86]:

```
#    results
# results = []
```

```
pre = cb_preprocessor
pre.fit(X_train, y_train)

X_train_t = pre.transform(X_train)
X_test_t = pre.transform(X_test)

best_cb_model = CatBoostClassifier(
    loss_function="Logloss",
    eval_metric="AUC",
    random_seed=42,
    verbose=False,
    cat_features=cat_idx,
    **study_cb.best_params
)

best_cb_model.fit(
    X_train_t, y_train,
    eval_set=(X_test_t, y_test),
    early_stopping_rounds=100
)

y_scores = best_cb_model.predict_proba(X_test_t)[:, 1]
y_pred  = best_cb_model.predict(X_test_t)

results_catboost = {
    "Model": "Tuned CatBoost",
    "Accuracy": accuracy_score(y_test, y_pred),
    "Precision": precision_score(y_test, y_pred, zero_division=0),
    "Recall": recall_score(y_test, y_pred, zero_division=0),
    "F1": f1_score(y_test, y_pred, zero_division=0),
    "ROC-AUC": roc_auc_score(y_test, y_scores),
}

results.append(results_catboost)
```

[89]:

```
pd.DataFrame(results)
```

[89]:

```
                 Model  Accuracy  Precision    Recall        F1   ROC-AUC
0  Logistic Regression  0.805536   0.657233  0.558824  0.604046  0.841874
1            SVM (RBF)  0.791341   0.641844  0.483957  0.551829  0.790491
2            KNN (k=7)  0.764372   0.554404  0.572193  0.563158  0.804810
3        Random Forest  0.776437   0.600683  0.470588  0.527736  0.819121
4             CatBoost  0.805536   0.676056  0.513369  0.583587  0.844270
5        LogReg (tuned)  0.804826   0.655172  0.558824  0.603175  0.841169
6       Tuned CatBoost  0.800568   0.656566  0.521390  0.581222  0.846425
7       Tuned CatBoost  0.808375   0.673333  0.540107  0.599407  0.846924
```

[90]:

```python
def optimize_threshold(trial):
    thr = trial.suggest_float("threshold", 0.05, 0.95)

    y_pred = (y_scores >= thr).astype(int)

    prec = precision_score(y_test, y_pred, zero_division=0)
    rec  = recall_score(y_test, y_pred, zero_division=0)
    f1   = f1_score(y_test, y_pred, zero_division=0)

    #       F1-    +
    score = f1 + 0.1 * prec + 0.1 * rec

    return score


study_thr = optuna.create_study(direction="maximize")
study_thr.optimize(optimize_threshold, n_trials=100)

study_thr.best_params
```

```
[I 2025-11-21 15:12:34,628] A new study created in memory with name: no-
name-7a439bee-69bc-403a-95c1-03f88253ac4f
[I 2025-11-21 15:12:34,652] Trial 0 finished with value: 0.7135055558715164 and
parameters: {'threshold': 0.11685412729686837}. Best is trial 0 with value:
0.7135055558715164.
[I 2025-11-21 15:12:34,674] Trial 1 finished with value: 0.6601793180421639 and
parameters: {'threshold': 0.5577852368603629}. Best is trial 0 with value:
0.7135055558715164.
[I 2025-11-21 15:12:34,697] Trial 2 finished with value: 0.6669626505707628 and
parameters: {'threshold': 0.05860960090985782}. Best is trial 0 with value:
0.7135055558715164.
[I 2025-11-21 15:12:34,717] Trial 3 finished with value: 0.1111730572306292 and
parameters: {'threshold': 0.9175107050922531}. Best is trial 0 with value:
0.7135055558715164.
[I 2025-11-21 15:12:34,735] Trial 4 finished with value: 0.7220351592757475 and
parameters: {'threshold': 0.15395130861953726}. Best is trial 4 with value:
0.7220351592757475.
[I 2025-11-21 15:12:34,753] Trial 5 finished with value: 0.7500976659583882 and
parameters: {'threshold': 0.2511200852040709}. Best is trial 5 with value:
0.7500976659583882.
[I 2025-11-21 15:12:34,770] Trial 6 finished with value: 0.7346228323659136 and
parameters: {'threshold': 0.18318689434987068}. Best is trial 5 with value:
0.7500976659583882.
[I 2025-11-21 15:12:34,789] Trial 7 finished with value: 0.7597079391197038 and
parameters: {'threshold': 0.37339999023427917}. Best is trial 7 with value:
0.7597079391197038.
[I 2025-11-21 15:12:34,807] Trial 8 finished with value: 0.22271786417916448 and
```

```
parameters: {'threshold': 0.8314907917681222}. Best is trial 7 with value:
0.7597079391197038.
[I 2025-11-21 15:12:34,824] Trial 9 finished with value: 0.7542273294648217 and
parameters: {'threshold': 0.39075371460325997}. Best is trial 7 with value:
0.7597079391197038.
[I 2025-11-21 15:12:34,847] Trial 10 finished with value: 0.571718629360075 and
parameters: {'threshold': 0.6077385047976682}. Best is trial 7 with value:
0.7597079391197038.
[I 2025-11-21 15:12:34,870] Trial 11 finished with value: 0.7594850410333979 and
parameters: {'threshold': 0.3748614287775843}. Best is trial 7 with value:
0.7597079391197038.
[I 2025-11-21 15:12:34,895] Trial 12 finished with value: 0.7448246141958109 and
parameters: {'threshold': 0.40879901978409955}. Best is trial 7 with value:
0.7597079391197038.
[I 2025-11-21 15:12:34,917] Trial 13 finished with value: 0.7652491441867983 and
parameters: {'threshold': 0.33719432709151986}. Best is trial 13 with value:
0.7652491441867983.
[I 2025-11-21 15:12:34,939] Trial 14 finished with value: 0.4663168050468765 and
parameters: {'threshold': 0.6919791403148126}. Best is trial 13 with value:
0.7652491441867983.
[I 2025-11-21 15:12:34,961] Trial 15 finished with value: 0.75231290442989 and
parameters: {'threshold': 0.2993404130092395}. Best is trial 13 with value:
0.7652491441867983.
[I 2025-11-21 15:12:34,983] Trial 16 finished with value: 0.721688945308393 and
parameters: {'threshold': 0.49081510509610454}. Best is trial 13 with value:
0.7652491441867983.
[I 2025-11-21 15:12:35,009] Trial 17 finished with value: 0.7516698836165777 and
parameters: {'threshold': 0.2912478246912531}. Best is trial 13 with value:
0.7652491441867983.
[I 2025-11-21 15:12:35,032] Trial 18 finished with value: 0.47098728090238096
and parameters: {'threshold': 0.686321089270544}. Best is trial 13 with value:
0.7652491441867983.
[I 2025-11-21 15:12:35,053] Trial 19 finished with value: 0.7441630850136686 and
parameters: {'threshold': 0.44038984813741033}. Best is trial 13 with value:
0.7652491441867983.
[I 2025-11-21 15:12:35,077] Trial 20 finished with value: 0.7468671855177988 and
parameters: {'threshold': 0.23566305721371153}. Best is trial 13 with value:
0.7652491441867983.
[I 2025-11-21 15:12:35,101] Trial 21 finished with value: 0.7653046884593685 and
parameters: {'threshold': 0.35212902830297127}. Best is trial 21 with value:
0.7653046884593685.
[I 2025-11-21 15:12:35,125] Trial 22 finished with value: 0.7663261044953519 and
parameters: {'threshold': 0.33277007517503965}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,153] Trial 23 finished with value: 0.758343359197968 and
parameters: {'threshold': 0.3162293952762993}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,176] Trial 24 finished with value: 0.7091219070062539 and
```

```
parameters: {'threshold': 0.5164034956686969}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,197] Trial 25 finished with value: 0.7218859817317493 and
parameters: {'threshold': 0.47498893491554534}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,226] Trial 26 finished with value: 0.7419723405017523 and
parameters: {'threshold': 0.20774425642501462}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,284] Trial 27 finished with value: 0.7575322126844863 and
parameters: {'threshold': 0.31308599991648783}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,319] Trial 28 finished with value: 0.6276862540128155 and
parameters: {'threshold': 0.5791299285833059}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,345] Trial 29 finished with value: 0.7081106713880219 and
parameters: {'threshold': 0.10893617658719207}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,386] Trial 30 finished with value: 0.7588349174915253 and
parameters: {'threshold': 0.3646126861992251}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,432] Trial 31 finished with value: 0.7584320464988986 and
parameters: {'threshold': 0.3569024341663302}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,456] Trial 32 finished with value: 0.7421688963708485 and
parameters: {'threshold': 0.4381749736085038}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,479] Trial 33 finished with value: 0.7515309197775456 and
parameters: {'threshold': 0.2559951631569415}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,500] Trial 34 finished with value: 0.7640512128428225 and
parameters: {'threshold': 0.34820357204648295}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,522] Trial 35 finished with value: 0.7081106713880219 and
parameters: {'threshold': 0.10890485279587214}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,545] Trial 36 finished with value: 0.6798769701902224 and
parameters: {'threshold': 0.5347858360107894}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,567] Trial 37 finished with value: 0.7227718360071301 and
parameters: {'threshold': 0.15573633441551182}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,589] Trial 38 finished with value: 0.7400243827848484 and
parameters: {'threshold': 0.45320327417887357}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,613] Trial 39 finished with value: 0.7516527140352012 and
parameters: {'threshold': 0.21724917200664828}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,635] Trial 40 finished with value: 0.7521000243971476 and
```

```
parameters: {'threshold': 0.2689335462743563}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,658] Trial 41 finished with value: 0.7630842960351176 and
parameters: {'threshold': 0.33932577243385964}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,685] Trial 42 finished with value: 0.7597062308892572 and
parameters: {'threshold': 0.32700354588859903}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,709] Trial 43 finished with value: 0.7429454390451833 and
parameters: {'threshold': 0.4054694128405141}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,731] Trial 44 finished with value: 0.7663261044953519 and
parameters: {'threshold': 0.33327410037715666}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,753] Trial 45 finished with value: 0.6596758802553484 and
parameters: {'threshold': 0.05078703219784453}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,776] Trial 46 finished with value: 0.0 and parameters:
{'threshold': 0.9392697411344801}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,800] Trial 47 finished with value: 0.7290509003428108 and
parameters: {'threshold': 0.16927686120778657}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,823] Trial 48 finished with value: 0.7441395138986859 and
parameters: {'threshold': 0.4035449715676013}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,848] Trial 49 finished with value: 0.7480466743384327 and
parameters: {'threshold': 0.2800651972529995}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,870] Trial 50 finished with value: 0.7388081434666804 and
parameters: {'threshold': 0.19856695098845215}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,891] Trial 51 finished with value: 0.7663261044953519 and
parameters: {'threshold': 0.3333496509809824}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,916] Trial 52 finished with value: 0.760425917828994 and
parameters: {'threshold': 0.35642686981145577}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,942] Trial 53 finished with value: 0.7452941322300792 and
parameters: {'threshold': 0.4292306904952275}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,965] Trial 54 finished with value: 0.7479595120700616 and
parameters: {'threshold': 0.24609106881816672}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:35,987] Trial 55 finished with value: 0.19664956891847651
and parameters: {'threshold': 0.8526703732605443}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,013] Trial 56 finished with value: 0.7515309505100843 and
```

```
parameters: {'threshold': 0.2985856705224138}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,036] Trial 57 finished with value: 0.760676691748065 and
parameters: {'threshold': 0.38775692797740013}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,058] Trial 58 finished with value: 0.726066847787673 and
parameters: {'threshold': 0.4656794632714144}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,079] Trial 59 finished with value: 0.5221964518585295 and
parameters: {'threshold': 0.649546409854223}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,101] Trial 60 finished with value: 0.7663261044953519 and
parameters: {'threshold': 0.3337459360930574}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,125] Trial 61 finished with value: 0.7597062308892572 and
parameters: {'threshold': 0.3268699038242207}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,147] Trial 62 finished with value: 0.7480466743384327 and
parameters: {'threshold': 0.28067010730218606}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,170] Trial 63 finished with value: 0.7603973027352393 and
parameters: {'threshold': 0.3756695085159267}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,208] Trial 64 finished with value: 0.7630842960351176 and
parameters: {'threshold': 0.33973775825226005}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,250] Trial 65 finished with value: 0.7416665047828808 and
parameters: {'threshold': 0.4238707101697846}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,274] Trial 66 finished with value: 0.7482968065269546 and
parameters: {'threshold': 0.24369835131142392}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,320] Trial 67 finished with value: 0.7229864062769709 and
parameters: {'threshold': 0.5014127577528869}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,366] Trial 68 finished with value: 0.7617719691535769 and
parameters: {'threshold': 0.3556515907506062}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,393] Trial 69 finished with value: 0.7517128340409878 and
parameters: {'threshold': 0.3038441033600262}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,419] Trial 70 finished with value: 0.7506390615024359 and
parameters: {'threshold': 0.22099064216062375}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,445] Trial 71 finished with value: 0.7652491441867983 and
parameters: {'threshold': 0.33552825859169505}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,473] Trial 72 finished with value: 0.7579030537409612 and
```

```
parameters: {'threshold': 0.38565625782058255}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,495] Trial 73 finished with value: 0.7521000243971476 and
parameters: {'threshold': 0.26872755081500144}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,523] Trial 74 finished with value: 0.758343359197968 and
parameters: {'threshold': 0.3142575115345008}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,552] Trial 75 finished with value: 0.7402789343701295 and
parameters: {'threshold': 0.4165975383090226}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,586] Trial 76 finished with value: 0.7652491441867983 and
parameters: {'threshold': 0.33602024303550265}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,609] Trial 77 finished with value: 0.7146263249905381 and
parameters: {'threshold': 0.13284575630459156}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,629] Trial 78 finished with value: 0.753215113956802 and
parameters: {'threshold': 0.29347370052914024}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,649] Trial 79 finished with value: 0.7597079391197038 and
parameters: {'threshold': 0.3730324527870485}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,668] Trial 80 finished with value: 0.7211073353061184 and
parameters: {'threshold': 0.48349211437489276}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,694] Trial 81 finished with value: 0.7630842960351176 and
parameters: {'threshold': 0.34076465498240327}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,726] Trial 82 finished with value: 0.7639247113548626 and
parameters: {'threshold': 0.34329923090896364}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,778] Trial 83 finished with value: 0.7509715923485218 and
parameters: {'threshold': 0.39668150183420803}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,829] Trial 84 finished with value: 0.7400243827848484 and
parameters: {'threshold': 0.4519228149188659}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,887] Trial 85 finished with value: 0.7580717048636881 and
parameters: {'threshold': 0.3195446883414715}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,968] Trial 86 finished with value: 0.750956440373335 and
parameters: {'threshold': 0.26595181010352065}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:36,998] Trial 87 finished with value: 0.7585990882407969 and
parameters: {'threshold': 0.3671700342073786}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:37,057] Trial 88 finished with value: 0.7325771995820851 and
```

```
parameters: {'threshold': 0.18553420186239084}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:37,113] Trial 89 finished with value: 0.7530967276796952 and
parameters: {'threshold': 0.30294927362321766}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:37,159] Trial 90 finished with value: 0.7482968065269546 and
parameters: {'threshold': 0.24147109970416575}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:37,214] Trial 91 finished with value: 0.7639247113548626 and
parameters: {'threshold': 0.34363573121895685}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:37,259] Trial 92 finished with value: 0.7597062308892572 and
parameters: {'threshold': 0.32412769950439513}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:37,290] Trial 93 finished with value: 0.7573108669155825 and
parameters: {'threshold': 0.35978505367082625}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:37,341] Trial 94 finished with value: 0.7480466743384327 and
parameters: {'threshold': 0.28077416229393953}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:37,398] Trial 95 finished with value: 0.7441395138986859 and
parameters: {'threshold': 0.4040676856954322}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:37,447] Trial 96 finished with value: 0.7630842960351176 and
parameters: {'threshold': 0.3417713894801406}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:37,509] Trial 97 finished with value: 0.7579030537409612 and
parameters: {'threshold': 0.3835512123282778}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:37,561] Trial 98 finished with value: 0.7522836198408835 and
parameters: {'threshold': 0.2857802122405756}. Best is trial 22 with value:
0.7663261044953519.
[I 2025-11-21 15:12:37,606] Trial 99 finished with value: 0.758887957710403 and
parameters: {'threshold': 0.32057825203798207}. Best is trial 22 with value:
0.7663261044953519.
```

[90]:
```
{'threshold': 0.33277007517503965}
```

[106]:
```python
 #
best_thr = study_thr.best_params["threshold"]

y_pred_thr = (y_scores >= best_thr).astype(int)

results_catboost_thr = {
    "Model": "Tuned CatBoost + Tuned Threshold",
    "Accuracy": accuracy_score(y_test, y_pred_thr),
    "Precision": precision_score(y_test, y_pred_thr, zero_division=0),
    "Recall": recall_score(y_test, y_pred_thr, zero_division=0),
    "F1 Score": f1_score(y_test, y_pred_thr, zero_division=0),
    "ROC AUC": roc_auc_score(y_test, y_scores)
}

results.append(results_catboost_thr)

pd.DataFrame(results)
```

[106]:

```
                                Model  Accuracy  Precision    Recall        F1  \
0                 Logistic Regression  0.805536   0.657233  0.558824  0.604046
1                           SVM (RBF)  0.791341   0.641844  0.483957  0.551829
2                           KNN (k=7)  0.764372   0.554404  0.572193  0.563158
3                       Random Forest  0.776437   0.600683  0.470588  0.527736
4                            CatBoost  0.805536   0.676056  0.513369  0.583587
5                      LogReg (tuned)  0.804826   0.655172  0.558824  0.603175
7                      Tuned CatBoost  0.808375   0.673333  0.540107  0.599407
9  Tuned CatBoost + Tuned Threshold  0.773598   0.554672  0.745989  0.636260

    ROC-AUC
0  0.841874
1  0.790491
2  0.804810
3  0.819121
4  0.844270
5  0.841169
7  0.846924
9  0.846924
```

[109]:

```
# 1)           - CatBoost                 ,            X_train
feature_names = X_train.columns.tolist()

# 2)
fi = pd.DataFrame({
    "Feature": feature_names,
    "Baseline CatBoost Importance": cb_model.get_feature_importance(),
    "Tuned CatBoost Importance": best_cb_model.get_feature_importance(),
})

fi
```

[109]:

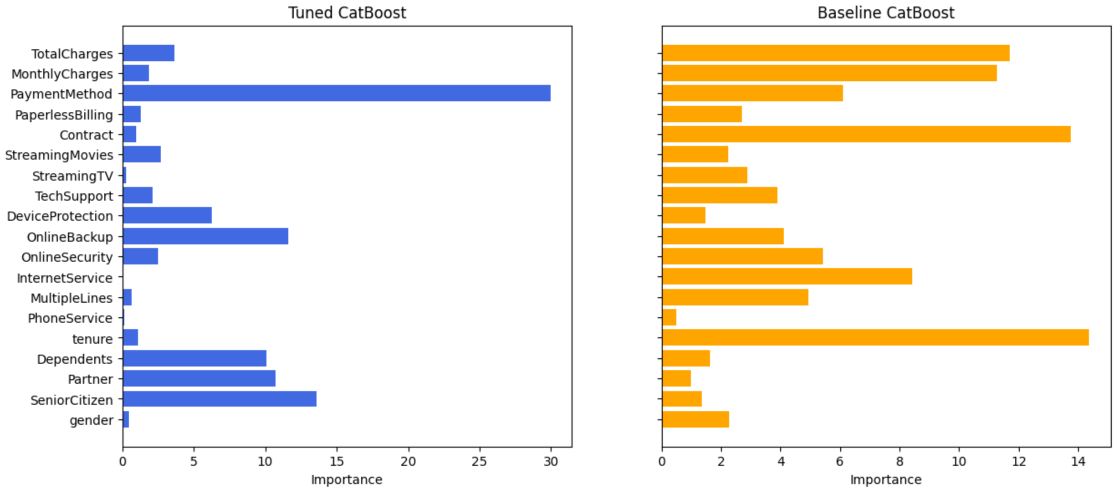
```
            Feature  Baseline CatBoost Importance  Tuned CatBoost Importance
0            gender                      2.263302                   0.471446
1     SeniorCitizen                      1.348421                  13.568037
2           Partner                      0.982945                  10.696227
3        Dependents                      1.622667                  10.103402
4            tenure                     14.373637                   1.121126
```

```
5        PhoneService           0.485607          0.116135
6       MultipleLines           4.932329          0.617744
7     InternetService           8.435215          0.085294
8      OnlineSecurity           5.416715          2.485342
9        OnlineBackup           4.107117         11.642309
10   DeviceProtection           1.470028          6.263567
11        TechSupport           3.892558          2.094057
12        StreamingTV           2.887360          0.296493
13    StreamingMovies           2.242390          2.700191
14           Contract          13.756544          0.960772
15   PaperlessBilling           2.697946          1.306437
16      PaymentMethod           6.112204         29.947958
17     MonthlyCharges          11.275231          1.858162
18       TotalCharges          11.697784          3.665302
```

[113]:

```
fig, axes = plt.subplots(1, 2, figsize=(14, 6), sharey=True)
axes[0].barh(fi["Feature"], fi["Tuned CatBoost Importance"], color="royalblue")
axes[0].set_title("Tuned CatBoost")
axes[0].set_xlabel("Importance")

axes[1].barh(fi["Feature"], fi["Baseline CatBoost Importance"], color="orange")
axes[1].set_title("Baseline CatBoost")
axes[1].set_xlabel("Importance")
plt.show()
```


Tuned CatBoost
Baseline CatBoost
TotalCharges
MonthlyCharges
PaymentMethod
PaperlessBilling
Contract
StreamingMovies
StreamingTV
TechSupport
DeviceProtection
OnlineBackup
OnlineSecurity
InternetService
MultipleLines
PhoneService
tenure
Dependents
Partner
SeniorCitizen
gender
Importance
Importance