In [ ]:
```python
import numpy as np
import pandas as pd
import xgboost as xgb
import lightgbm as lgb
from catboost import CatBoostClassifier
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import seaborn as sns

import os

for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

In [ ]:
```python
url = "https://raw.githubusercontent.com/erkansirin78/datasets/master/Churn_Modelling.csv"
origin = pd.read_csv(url)
train = pd.read_csv('/kaggle/input/knou-deeplearning2025-finalterm/train.csv')
test = pd.read_csv('/kaggle/input/knou-deeplearning2025-finalterm/test.csv')
sample_submission = pd.read_csv('/kaggle/input/knou-deeplearning2025-finalterm/sample_submission.csv')
```

In [ ]:
```python
train_drop = train.drop(['id'], axis=1)
origin_drop = origin.drop(['RowNumber'], axis=1)
combined = pd.concat([train_drop, origin_drop], axis=0)
```

In [ ]:
```python
# 데이터 시각화
plt.figure(figsize=(15, 10))

plt.subplot(2, 3, 1)
combined['Exited'].value_counts().plot(kind='bar', color=['green', 'red'])
plt.title('Customer Churn Distribution')
plt.xlabel('Exited')
plt.ylabel('Count')

plt.subplot(2, 3, 2)
pd.crosstab(combined['Geography'], combined['Exited'], normalize='index').plot(kind='bar')
plt.title('Churn Rate by Geography')
plt.xlabel('Geography')
```

```
plt.ylabel('Rate')

plt.subplot(2, 3, 3)
pd.crosstab(combined['Gender'], combined['Exited'], normalize='index').plot(kind='bar')
plt.title('Churn Rate by Gender')
plt.xlabel('Gender')
plt.ylabel('Rate')

plt.subplot(2, 3, 4)
plt.hist([combined[combined['Exited']==0]['Age'], combined[combined['Exited']==1]['Age']], bins=20, label=['Stayed'
plt.title('Age Distribution')
plt.xlabel('Age')
plt.legend()

plt.subplot(2, 3, 5)
plt.hist([combined[combined['Exited']==0]['Balance'], combined[combined['Exited']==1]['Balance']], bins=20, label=[
plt.title('Balance Distribution')
plt.xlabel('Balance')
plt.legend()

plt.subplot(2, 3, 6)
numeric_cols = combined.select_dtypes(include=[np.number]).columns
sns.heatmap(combined[numeric_cols].corr(), annot=True, fmt='.2f', cmap='coolwarm')
plt.title('Correlation Heatmap')

plt.tight_layout()
plt.show()
```

In [ ]:

```
# Geography 인코딩
combined['Geography_France'] = (combined['Geography'] == 'France').astype(int)
combined['Geography_Germany'] = (combined['Geography'] == 'Germany').astype(int)
combined['Geography_Spain'] = (combined['Geography'] == 'Spain').astype(int)

# Gender 인코딩
combined['Gender_Female'] = (combined['Gender'] == 'Female').astype(int)

# Surname 빈도수
surname_freq = combined['Surname'].value_counts().to_dict()
combined['SurnameFrequency'] = combined['Surname'].map(surname_freq)
```

```
X = combined.drop(['Surname', 'Geography', 'Gender', 'Exited'], axis=1)
y = combined.Exited

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=22)
```

In [ ]:

```
# XGBoost
dtrain = xgb.DMatrix(X_train, label=y_train)
dtest = xgb.DMatrix(X_test, label=y_test)

xgb_model = xgb.train(
    {
        'objective': 'binary:logistic',
        'eval_metric': 'logloss',
        'max_depth': 6,
        'learning_rate': 0.05,
        'subsample': 0.8,
        'colsample_bytree': 0.8,
    },
    dtrain,
    num_boost_round=200,
)
```

In [ ]:

```
# LightGBM
lgb_train = lgb.Dataset(X_train, y_train)

lgb_model = lgb.train(
    {
        'objective': 'binary',
        'metric': 'binary_logloss',
        'num_leaves': 31,
        'learning_rate': 0.05,
        'feature_fraction': 0.8,
        'bagging_fraction': 0.8,
        'bagging_freq': 5,
        'verbose': -1
    },
    lgb_train,
    num_boost_round=200,
)
```

In [ ]:
```
# CatBoost
cat_model = CatBoostClassifier(
    iterations=200,
    learning_rate=0.05,
    depth=6,
    verbose=0
)

cat_model.fit(X_train, y_train)
```

In [ ]:
```
# 테스트 데이터 전처리
test_copy = test.copy()

test_copy['Geography_France'] = (test_copy['Geography'] == 'France').astype(int)
test_copy['Geography_Germany'] = (test_copy['Geography'] == 'Germany').astype(int)
test_copy['Geography_Spain'] = (test_copy['Geography'] == 'Spain').astype(int)
test_copy['Gender_Female'] = (test_copy['Gender'] == 'Female').astype(int)
test_copy['SurnameFrequency'] = test_copy['Surname'].map(surname_freq).fillna(1)

test_drop = test_copy.drop(['id', 'Surname', 'Geography', 'Gender'], axis=1)
```

In [ ]:
```
# 예측
dtest_final = xgb.DMatrix(test_drop)
xgb_pred = xgb_model.predict(dtest_final)
lgb_pred = lgb_model.predict(test_drop)
cat_pred = cat_model.predict_proba(test_drop)[:, 1]

# 앙상블
y_pred = (xgb_pred + lgb_pred + cat_pred) / 3
```

In [ ]:
```
sample_submission['Exited'] = y_pred
sample_submission.to_csv('submission.csv', index=False)
```