Home Credit Default Risk (HCDR)

Vinay Reddy

- Machine Learning/Analytics
- Process Improvement/Automation
- Financial Services/Banking
- Enterprise Data Management

Add meaning picture from phase 1

Home Credit Default Risk (HCDR) Kaggle Competition. The goal of the projects is to predict whether or not a client will repay a loan. We use a home credit dataset with a large number of features to predict if a client will be able to repay a loan. This is a standard supervised classification task. Home Credit makes use of a variety of alternative data, including rents and transactional information to predict their clients' repayment abilities.

Some of the challenges

- Imbalanced
- Class imbalanced: over-represented and relevance of rare cases
- 2.71% of data are positive(1)
- Dealing with missing data
- Handling categorical data
- Incorporating heterogeneous data

Kaggle API setup

Kaggle is a Data Science Competition Platform which shares a lot of datasets. In the past, it was inadvisable to submit your result or your team's rank through this current is an known and long past time Below that you can connect with Kaggle via the command line.

```
! pip install -q kaggle
! mkdir ~/.kaggle
! echo '{"username":"xxxxxx","key":"xxxxxxxxxxxxxx"}' > ~/.kaggle/kaggle.json
! chmod 600 ~/.kaggle/kaggle.json
! kaggle competitions list -s home-credit-default-risk
```

In order to use the kaggle, it takes the key-value JSON for authentication.

```
! ls ~/.kaggle
```

Output: 'kaggle.json' means that the kaggle.json file is in place.

```
! kaggle competitions download -c home-credit-default-risk
```

Output: Downloading home-credit-default-risk.zip to /content

```
! ls -lrtha *.zip
```

Output: -rw-r--r-- 1 root root 688M Jun 11 17:42 home-credit-default-risk.zip

```
! unzip home-credit-default-risk.zip -d home-credit-default-risk
```

Output: Archive: home-credit-default-risk.zip
inflating: home-credit-default-risk/sample_submission.csv.7z
inflating: home-credit-default-risk/application_train.csv.7z
inflating: home-credit-default-risk/application_test.csv.7z
inflating: home-credit-default-risk/bureau.zip
inflating: home-credit-default-risk/bureau_balance.zip
inflating: home-credit-default-risk/POS_CASH_balance.zip
inflating: home-credit-default-risk/credit_card_balance.zip
inflating: home-credit-default-risk/previous_application.zip
inflating: home-credit-default-risk/installments_payments.zip
inflating: home-credit-default-risk/HCDR.ipynb

```
! ls home
```

Output: home_credit_default_risk

```
! ls -lrtha home/home-credit-default-risk
```

Output: drwxr-xr-x 2 root root 4096 Jun 11 17:42 home/home-credit-default-risk
-rw-r--r-- 1 root root 688M Jun 11 17:42 home-credit-default-risk.zip
drwxr-xr-x 1 root root 4096 Jun 11 17:43 sample_data
drwxr-xr-x 3 root root 4096 Jun 11 17:43 home-credit-default-risk
drwxr-xr-x 1 root root 4096 Jun 11 17:43 .kaggle
drwxr-xr-x 1 root root 4096 Jun 11 17:43 .config
drwxr-xr-x 1 root root 4096 Jun 11 17:43 .ipynb_checkpoints
drwxr-xr-x 1 root root 4096 Jun 11 17:43 home

```
! ls home/home-credit-default-risk
```

Output: application_test.csv.7z
application_train.csv.7z
bureau.zip
bureau_balance.zip
credit_card_balance.zip
HCDR.ipynb
installments_payments.zip
POS_CASH_balance.zip
previous_application.zip
sample_submission.csv.7z

Dataset and how to download

Back ground Home Credit Group

Many people struggle to get loans due to insufficient or non-existent credit histories. Unfortunately, this unfortunate population is often taken advantage of by predatory lenders.

Home Credit Group

Home Credit strives to broaden financial inclusion for the unbanked population by providing a positive and safe borrowing experience. In order to make this possible, they utilize various alternative data and machine learning methods to assess clients' credit worthiness and to tailor products, including rates and transactional information, to predict their clients' repayment abilities.

While financial institutions have been using machine learning and other statistical models to assess zero- or low-credit clients, there is still plenty of room for improvement. There is a great deal of potential for more powerful predictive models to deploy more responsibly lending to the unbanked population, and newer data sources and techniques can be explored. Home Credit is hosting this competition to see what kind of models the machine learning and data science community can develop to help broaden financial inclusion, to unveil their clients' repayment abilities.

Background on Dataset

Home Credit is not the first company to utilize alternative data to predict clients' repayment abilities. In fact, many credit bureaus are starting to incorporate more alternative data into their credit models. For example, Experian plans to add 1 billion pieces of rent payment data to its credit bureau database. Alternative data can be a powerful tool for lenders looking to extend responsible credit to the unbanked population.

While Home Credit is currently using various statistical and machine learning methods to make these predictions, they're challenging Kaggle competitors to see what kind of models they should be deploying in the future. Kaggle competitors will have access to a wide range of demographic, housing, financial, and transactional data from hundreds of thousands of clients to inform their models.

Data files overview

There are 7 different sources of data:

```
! ls home/home-credit-default-risk
```

Output: application_test.csv.7z
application_train.csv.7z
bureau.zip
bureau_balance.zip
credit_card_balance.zip
HCDR.ipynb
installments_payments.zip
POS_CASH_balance.zip
previous_application.zip
sample_submission.csv.7z

```
! ls -lrtha home/home-credit-default-risk/*.zip
```

Output: -rw-r--r-- 1 root root 688M Jun 11 17:42 home/home-credit-default-risk/home-credit-default-risk.zip

```
import os, sys
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
sns.set_style("whitegrid")
sns.set_palette("colorblind")
pd.options.mode.chained_assignment = None
pd.options.display.max_columns = 50
%matplotlib inline
path = "home/home-credit-default-risk"
print("Data files: {}".format(os.listdir(path)))
def get_filesize(p):
    return "{:.2f}MB*.format(os.path.getsize(p)/(1024*1024))"
print("Application train: {}".format(get_filesize(path+"/application_train.csv.7z")))
print("Application test: {}".format(get_filesize(path+"/application_test.csv.7z")))
print("Bureau: {}".format(get_filesize(path+"/bureau.zip")))
print("Bureau balance: {}".format(get_filesize(path+"/bureau_balance.zip")))
print("Credit card balance: {}".format(get_filesize(path+"/credit_card_balance.zip")))
print("Installments payments: {}".format(get_filesize(path+"/installments_payments.zip")))
print("POS cash balance: {}".format(get_filesize(path+"/POS_CASH_balance.zip")))
print("Previous application: {}".format(get_filesize(path+"/previous_application.zip")))
print("Sample submission: {}".format(get_filesize(path+"/sample_submission.csv.7z")))
```

Output: Data files: ['sample_submission.csv.7z', 'POS_CASH_balance.zip', 'previous_application.zip', 'installments_payments.zip', 'credit_card_balance.zip', 'bureau_balance.zip', 'bureau.zip', 'application_train.csv.7z', 'application_test.csv.7z', 'HCDR.ipynb']
Application train: 688.00MB*
Application test: 688.00MB*
Bureau: 688.00MB*
Bureau balance: 688.00MB*
Credit card balance: 688.00MB*
Installments payments: 688.00MB*
POS cash balance: 688.00MB*
Previous application: 688.00MB*
Sample submission: 688.00MB*

```
! 7z e home/home-credit-default-risk/application_train.csv.7z -d home/home-credit-default-risk
! 7z e home/home-credit-default-risk/application_test.csv.7z -d home/home-credit-default-risk
! 7z e home/home-credit-default-risk/bureau.zip -d home/home-credit-default-risk
! 7z e home/home-credit-default-risk/bureau_balance.zip -d home/home-credit-default-risk
! 7z e home/home-credit-default-risk/credit_card_balance.zip -d home/home-credit-default-risk
! 7z e home/home-credit-default-risk/installments_payments.zip -d home/home-credit-default-risk
! 7z e home/home-credit-default-risk/POS_CASH_balance.zip -d home/home-credit-default-risk
! 7z e home/home-credit-default-risk/previous_application.zip -d home/home-credit-default-risk
! 7z e home/home-credit-default-risk/sample_submission.csv.7z -d home/home-credit-default-risk
```

Helper functions for EDA

```
def draw_missing_data_table(df):
    total = df.isnull().sum().sort_values(ascending=False)
    percent = (df.isnull()/df.shape[0]).sum()*100.0).sort_values(ascending=False)
    ttype = df.dtypes[df.dtypes.sort_values(ascending=False).index]
    dtype = ttype.rename('Type')
    dtype = dtype.apply(lambda x: str(x))
    return np.round(pd.concat[total, percent, dtype], axis=1, keys=['Total', 'Percent', 'Type'])
def get_dtypes(df):
    return df.dtypes.value_counts()
def get_object_columns(df):
    return df[df.columns[df.dtypes[df.dtypes=='object'].index]].columns
def get_numeric_columns(df):
    return df[df.columns[df.dtypes[df.dtypes!='object'].index]].columns
def get_binary_columns(df):
    binary = []
    for col in df.columns:
        if df.shape[0] == df[col].sum(): continue
        if df.shape[0] == df[df[col]==1].shape[0]:
            print("{:30}{:>7}".format(col,df[df[col]==1].shape[0]))
            binary.append(col)
    return binary
def get_uniq_values(df):
    for col in df.columns:
        if df.shape[0] == df[col].nunique(): continue
        if df[col].dtype == 'object':
            print("{:30}{:>7}".format(col,df[col].nunique()))
            print(df[col].value_counts()[df[col].value_counts()>1].\
                  sort_values(ascending=False).\
                  to_string(dtype=False,sparsity=False,name=False),'\n')
        else:
            print("{:30}{:>7}".format(col,df[col].nunique()))
def get_corr(df,cols,thr=0.5):
    corr = df[cols].corr()
    c = corr.abs()
    s = c.unstack()
    so = s.sort_values(kind="quicksort")
    for i,v in zip(so[so>thr].index.values,so[so>thr].values):
        if i[0] != i[1]:
            print("{:30}{:30}{:>8}".format(i[0],i[1],np.round(v,4)))
def get_top_values(df,cols,n=5):
    for c in cols:
        print("{:30}{:>7}".format(c,df[c].nunique()))
        print(df[c].value_counts()[df[c].value_counts()>1][:n],'\n')
```

Exploratory Data Analysis

```
# Application train
app = pd.read_csv(path+'/application_train.csv')
app.shape
```

Output: (307511, 122)

```
app.sample(3)
```

Output:

| | SK_ID_CURR | TARGET | NAME_CONTRACT_TYPE | CODE_GENDER | FLAG_OWN_CAR | FLAG_OWN_REALTY | CNT_CHILDREN | AMT_INCOME_TOTAL | AMT_CREDIT | AMT_ANNUITY | AMT_GOODS_PRICE | NAME_TYPE_SUITE | NAME_INCOME_TYPE | NAME_EDUCATION_TYPE | NAME_FAMILY_STATUS | NAME_HOUSING_TYPE | REGION_POPULATION_RELATIVE | DAYS_BIRTH | DAYS_EMPLOYED | DAYS_REGISTRATION | DAYS_ID_PUBLISH | OWN_CAR_AGE | FLAG_MOBIL | FLAG_EMP_PHONE | FLAG_WORK_PHONE | FLAG_CONT_MOBILE | FLAG_PHONE | FLAG_EMAIL | OCCUPATION_TYPE | CNT_FAM_MEMBERS | REGION_RATING_CLIENT | REGION_RATING_CLIENT_W_CITY | WEEKDAY_APPR_PROCESS_START | HOUR_APPR_PROCESS_START | REG_REGION_NOT_LIVE_REGION | REG_REGION_NOT_WORK_REGION | LIVE_REGION_NOT_WORK_REGION | REG_CITY_NOT_LIVE_CITY | REG_CITY_NOT_WORK_CITY | LIVE_CITY_NOT_WORK_CITY | ORGANIZATION_TYPE | EXT_SOURCE_1 | EXT_SOURCE_2 | EXT_SOURCE_3 | APARTMENTS_AVG | BASEMENTAREA_AVG | YEARS_BEGINEXPLUATATION_AVG | YEARS_BUILD_AVG | COMMONAREA_AVG | ELEVATORS_AVG | ENTRANCES_AVG | FLOORSMAX_AVG | FLOORSMIN_AVG | LANDAREA_AVG | LIVINGAPARTMENTS_AVG | LIVINGAREA_AVG | NONLIVINGAPARTMENTS_AVG | NONLIVINGAREA_AVG | APARTMENTS_MODE | BASEMENTAREA_MODE | YEARS_BEGINEXPLUATATION_MODE | YEARS_BUILD_MODE | COMMONAREA_MODE | ELEVATORS_MODE | ENTRANCES_MODE | FLOORSMAX_MODE | FLOORSMIN_MODE | LANDAREA_MODE | LIVINGAPARTMENTS_MODE | LIVINGAREA_MODE | NONLIVINGAPARTMENTS_MODE | NONLIVINGAREA_MODE | APARTMENTS_MEDI | BASEMENTAREA_MEDI | YEARS_BEGINEXPLUATATION_MEDI | YEARS_BUILD_MEDI | COMMONAREA_MEDI | ELEVATORS_MEDI | ENTRANCES_MEDI | FLOORSMAX_MEDI | FLOORSMIN_MEDI | LANDAREA_MEDI | LIVINGAPARTMENTS_MEDI | LIVINGAREA_MEDI | NONLIVINGAPARTMENTS_MEDI | NONLIVINGAREA_MEDI | FONDKAPREMONT_MODE | HOUSETYPE_MODE | TOTALAREA_MODE | WALLSMATERIAL_MODE | EMERGENCYSTATE_MODE | DEF_30_CNT_SOCIAL_CIRCLE | DEF_30_CNT_SOCIAL_CIRCLE | DEF_60_CNT_SOCIAL_CIRCLE | DEF_60_CNT_SOCIAL_CIRCLE | DAYS_LAST_PHONE_CHANGE | FLAG_DOCUMENT_2 | FLAG_DOCUMENT_3 | FLAG_DOCUMENT_4 | FLAG_DOCUMENT_5 | FLAG_DOCUMENT_6 | FLAG_DOCUMENT_7 | FLAG_DOCUMENT_8 | FLAG_DOCUMENT_9 | FLAG_DOCUMENT_10 | FLAG_DOCUMENT_11 | FLAG_DOCUMENT_12 | FLAG_DOCUMENT_13 | FLAG_DOCUMENT_14 | FLAG_DOCUMENT_15 | FLAG_DOCUMENT_16 | FLAG_DOCUMENT_17 | FLAG_DOCUMENT_18 | FLAG_DOCUMENT_19 | FLAG_DOCUMENT_20 | FLAG_DOCUMENT_21 | AMT_REQ_CREDIT_BUREAU_HOUR | AMT_REQ_CREDIT_BUREAU_DAY | AMT_REQ_CREDIT_BUREAU_WEEK | AMT_REQ_CREDIT_BUREAU_MON | AMT_REQ_CREDIT_BUREAU_QRT | AMT_REQ_CREDIT_BUREAU_YEAR |
|---|---|---

```
%cd /content
%pwd
!unzip -q home-credit-default-risk.zip -d home_credit_data
!ls home_credit_data
```

```
/content
/content
application_train.csv  bureau.csv                         HOME_CREDIT_STATEMENT.csv
application_test.csv   bureau_balance.csv                 previous_application.csv
credit_card_balance.csv installments_payments.csv         sample_submission.csv
HOME_CREDIT_DESCRIPTION.csv POS_CASH_balance.csv
```

```
%pip install -q -U kaggle
%mkdir ~/.kaggle
%cp kaggle.json ~/.kaggle/
%chmod 600 ~/.kaggle/kaggle.json
%kaggle competitions download -c home-credit-default-risk
```

```
Collecting kagglehub[hf_xet]>=0.3.4
  Downloading kagglehub-0.3.13-py3-none-whl (67 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 67.5/67.5 kB 3.7 MB/s eta 0:00:00
Requirement already satisfied: packaging in /usr/local/lib/python3.12/dist-packages/packaging-25.0-py3-none-any.whl (20 kB)
Requirement already satisfied: pyyaml in /usr/local/lib/python3.12/dist-packages/pyyaml-6.0.3-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (806 kB)
Requirement already satisfied: requests in /usr/local/lib/python3.12/dist-packages/requests-2.32.4-py3-none-any.whl (64 kB)
Requirement already satisfied: tqdm in /usr/local/lib/python3.12/dist-packages/tqdm-4.67.1-py3-none-any.whl (78 kB)
Requirement already satisfied: hf-xet<1.2.0,>=1.1.3 in /usr/local/lib/python3.12/dist-packages/hf_xet-1.1.10-cp37-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (3.2 MB)
Requirement already satisfied: charset-normalizer<3,>=2 in /usr/local/lib/python3.12/dist-packages/charset_normalizer-2.0.12-py3-none-any.whl (41 kB)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.12/dist-packages/idna-2.9-py3-none-any.whl (58 kB)
Requirement already satisfied: urllib3<1.26,>=1.21.1 in /usr/local/lib/python3.12/dist-packages/urllib3-1.25.11-py2.py3-none-any.whl (127 kB)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages/certifi-2025.8.3-py3-none-any.whl (156 kB)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.12/dist-packages/typing_extensions-4.14.1-py3-none-any.whl (43 kB)
Requirement already satisfied: filelock>=3.18.0 in /usr/local/lib/python3.12/dist-packages/filelock-3.18.0-py3-none-any.whl (10 kB)
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.12/dist-packages/fsspec-2025.3.0-py3-none-any.whl (193 kB)
Requirement already satisfied: shutil>=1.7.0 in /usr/local/lib/python3.12/dist-packages/shutil-1.7.0-py3-none-any.whl (19 kB)
Requirement already satisfied: typing>=3.7.4 in /usr/local/lib/python3.12/dist-packages/typing-3.7.4.3-py3-none-any.whl (26 kB)
Requirement already satisfied: setuptools>=39.2.0 in /usr/local/lib/python3.12/dist-packages/setuptools-75.2.0-py3-none-any.whl (1.2 MB)
Requirement already satisfied: six>=1.10.0 in /usr/local/lib/python3.12/dist-packages/six-1.16.0-py3-none-any.whl (11 kB)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.12/dist-packages/python-dateutil-2.8.2-py3-none-any.whl (247 kB)
Requirement already satisfied: python-slugify>=8.0.4 in /usr/local/lib/python3.12/dist-packages/python-slugify-8.0.4-py3-none-any.whl (10 kB)
Requirement already satisfied: text-unidecode>=1.3.1 in /usr/local/lib/python3.12/dist-packages/text-unidecode-1.3-py3-none-any.whl (78 kB)
Requirement already satisfied: webencodings>=0.5.1 in /usr/local/lib/python3.12/dist-packages/webencodings-0.5.1-py3-none-any.whl (11 kB)
Requirement already satisfied: hf-xet<1.2.0,>=1.1.3 in /usr/local/lib/python3.12/dist-packages/hf_xet-1.1.10-cp37-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (3.2 MB)
Requirement already satisfied: charset-normalizer<3,>=2 in /usr/local/lib/python3.12/dist-packages/charset_normalizer-2.0.12-py3-none-any.whl (41 kB)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.12/dist-packages/idna-2.9-py3-none-any.whl (58 kB)
Requirement already satisfied: urllib3<1.26,>=1.21.1 in /usr/local/lib/python3.12/dist-packages/urllib3-1.25.11-py2.py3-none-any.whl (127 kB)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages/certifi-2025.8.3-py3-none-any.whl (156 kB)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.12/dist-packages/typing_extensions-4.14.1-py3-none-any.whl (43 kB)
Requirement already satisfied: filelock>=3.18.0 in /usr/local/lib/python3.12/dist-packages/filelock-3.18.0-py3-none-any.whl (10 kB)
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.12/dist-packages/fsspec-2025.3.0-py3-none-any.whl (193 kB)
Requirement already satisfied: shutil>=1.7.0 in /usr/local/lib/python3.12/dist-packages/shutil-1.7.0-py3-none-any.whl (19 kB)
Requirement already satisfied: typing>=3.7.4 in /usr/local/lib/python3.12/dist-packages/typing-3.7.4.3-py3-none-any.whl (26 kB)
Requirement already satisfied: setuptools>=39.2.0 in /usr/local/lib/python3.12/dist-packages/setuptools-75.2.0-py3-none-any.whl (1.2 MB)
Requirement already satisfied: six>=1.10.0 in /usr/local/lib/python3.12/dist-packages/six-1.16.0-py3-none-any.whl (11 kB)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.12/dist-packages/python-dateutil-2.8.2-py3-none-any.whl (247 kB)
Requirement already satisfied: python-slugify>=8.0.4 in /usr/local/lib/python3.12/dist-packages/python-slugify-8.0.4-py3-none-any.whl (10 kB)
Requirement already satisfied: text-unidecode>=1.3.1 in /usr/local/lib/python3.12/dist-packages/text-unidecode-1.3-py3-none-any.whl (78 kB)
Requirement already satisfied: webencodings>=0.5.1 in /usr/local/lib/python3.12/dist-packages/webencodings-0.5.1-py3-none-any.whl (11 kB)
```

```
%kaggle competitions download -c home-credit-default-risk
```

```
Dataset URL: https://www.kaggle.com/competitions/home-credit-default-risk
home-credit-default-risk.zip: Skipping, found more recently modified local copy (use --force to force download)
```

```
!ls -lh home-credit-default-risk.zip
```

```
-rw-r--r-- 1 root root 1.1G Aug 14 07:12 home-credit-default-risk.zip
```

```
!unzip -q home-credit-default-risk.zip -d home_credit_data
!ls home_credit_data
```

```
application_train.csv  bureau.csv                         HOME_CREDIT_STATEMENT.csv
application_test.csv   bureau_balance.csv                 previous_application.csv
credit_card_balance.csv installments_payments.csv         sample_submission.csv
HOME_CREDIT_DESCRIPTION.csv POS_CASH_balance.csv
```

Application Train EDA

```
%matplotlib inline
%cd /content/home_credit_data
!head application_train.csv
```

```
/content/home_credit_data
SK_ID_CURR,TARGET,NAME_CONTRACT_TYPE,CODE_GENDER,FLAG_OWN_CAR,FLAG_OWN_REALTY,CNT_CHILDREN,AMT_INCOME_TOTAL,AMT_CREDIT,AMT_ANNUITY,AMT_GOODS_PRICE,NAME_TYPE_SUITE,NAME_INCOME_TYPE,NAME_EDUCATION_TYPE,NAME_FAMILY_STATUS,NAME_HOUSING_TYPE,REGION_POPULATION_RELATIVE,DAYS_BIRTH,DAYS_EMPLOYED,DAYS_REGISTRATION,DAYS_ID_PUBLISH,OWN_CAR_AGE,FLAG_MOBIL,FLAG_EMP_PHONE,FLAG_WORK_PHONE,FLAG_CONT_MOBILE,FLAG_PHONE,FLAG_EMAIL,OCCUPATION_TYPE,CNT_FAM_MEMBERS,REGION_RATING_CLIENT,REGION_RATING_CLIENT_W_CITY,WEEKDAY_APPR_PROCESS_START,HOUR_APPR_PROCESS_START,REG_REGION_NOT_LIVE_REGION,REG_REGION_NOT_WORK_REGION,LIVE_REGION_NOT_WORK_REGION,REG_CITY_NOT_LIVE_CITY,REG_CITY_NOT_WORK_CITY,LIVE_CITY_NOT_WORK_CITY,ORGANIZATION_TYPE,EXT_SOURCE_1,EXT_SOURCE_2,EXT_SOURCE_3,APARTMENTS_AVG,BASEMENTAREA_AVG,YEARS_BEGINEXPLUATATION_AVG,YEARS_COMMON,ELEVATORS_AVG,ENTRANCES_AVG,NONLIVINGAPARTMENTS_AVG,NONLIVINGAREA_AVG,APARTMENTS_MODE,BASEMENTAREA_MODE,YEARS_BEGINEXPLUATATION_MODE,YEARS_COMMON,ELEVATORS_MODE,ENTRANCES_MODE,NONLIVINGAPARTMENTS_MODE,NONLIVINGAREA_MODE,APARTMENTS_MEDI,BASEMENTAREA_MEDI,YEARS_BEGINEXPLUATATION_MEDI,YEARS_COMMON,ELEVATORS_MEDI,ENTRANCES_MEDI,NONLIVINGAPARTMENTS_MEDI,NONLIVINGAREA_MEDI,FONDKAPREMONT_MODE,HOUSETYPE_MODE,TOTALAREA_MODE,WALLSMATERIAL_MODE,EMERGENCYSTATE_MODE,OBS_30_CNT,DEF_30_CNT,OBS_60_CNT,DEF_60_CNT,DAYS_DECISION,AMT_GOODS_PRICE,WEEKDAY_APPR_PROCESS_START,HOUR_APPR_PROCESS_START,FLAG_LAST_APPL_PER_CONTRACT,NFLAG_LAST_APPL_IN_DAYS,NAME_CONTRACT_STATUS,NAME_PAYMENT_TYPE,CODE_REJECT_REASON,NAME_TYPE_SUITE,NAME_CLIENT_TYPE,NAME_GOODS_CATEGORY,NAME_PORTFOLIO,NAME_PRODUCT_TYPE,CHANNEL_TYPE,SELLERPLACE_AREA,NAME_SELLER_INDUSTRY,CNT_INSTALMENT,CNT_INSTALMENT_FUTURE,NAME_YIELD_GROUP,PRODUCT_COMBINATION,DAYS_FIRST_DRAWING,DAYS_FIRST_DUE,DAYS_TERMINATION,NFLAG_INSURANCE,SK_ID_BUREAU,DAYS_CREDIT,CREDIT_DAY_OVERDUE,DAYS_CREDIT_ENDDATE,DAYS_ENDDATE_FACT,AMT_CREDIT_MAX_OVERDUE,CNT_CREDIT_PROLONG,AMT_CREDIT_SUM,AMT_CREDIT_SUM_DEBT,AMT_CREDIT_SUM_LIMIT,AMT_CREDIT_SUM_OVERDUE,CREDIT_TYPE,DAYS_CREDIT_UPDATE,AMT_ANNUITY,MONTHS_BALANCE,AMT_BALANCE,AMT_CREDIT_LIMIT_ACTUAL,AMT_DRAWINGS_ATM_CURRENT,AMT_DRAWINGS_CURRENT,AMT_DRAWINGS_OTHER_CURRENT,AMT_DRAWINGS_POS_CURRENT,AMT_INST_MIN_REGULARITY,AMT_PAYMENT_CURRENT,AMT_PAYMENT_TOTAL_CURRENT,AMT_RECEIVABLE_PRINCIPAL,AMT_RECEIVABLE,AMT_TOTAL_RECEIVABLE,CNT_DRAWINGS_ATM_CURRENT,CNT_DRAWINGS_CURRENT,CNT_DRAWINGS_OTHER_CURRENT,CNT_DRAWINGS_POS_CURRENT,CNT_INSTALMENT_MATURE_CUM,NAME_CONTRACT_STATUS,SK_DPD,SK_DPD_DEF,SK_ID_PREV,NUM_INSTALMENT_VERSION,NUM_INSTALMENT_NUMBER,DAYS_INSTALMENT,DAYS_ENTRY_PAYMENT,AMT_INSTALMENT,AMT_PAYMENT,
100002,1,Cash loans,M,N,Y,0,202500.0,406597.5,24700.5,351000.0,Unaccompanied,Working,Secondary / secondary special,Single / not married,House / apartment,0.018801,-9461,-637,-3648.0,-2120,,1,1,0,1,1,0,Laborers,1.0,2,2,WEDNESDAY,10,0,0,0,0,0,0,Business Entity 3,0.083037,0.262474,0.139376,0.0247,0.0369,0.9722,0.6192,0.0143,0.0000,0.0690,0.0833,0.0149,0.0714,0.0393,0.0000,0.0000,0.0000
```

```
  AMT_ID_PREV  NUM_INSTALMENT_NUMBER  AMT_BALANCE  AMT_CREDIT_LIMIT_ACTUAL  AMT_DRAWINGS_ATM_CURRENT  AMT_DR
count  1.5366e+06              1.5366e+06  1.5366e+06               1.5366e+06                  1.5366e+06  1.53
mean   1.5704e+06              3.3203e+01  5.8963e+04               1.5399e+05                  5.9540e+03  6.03
std    5.3260e+05              2.6644e+01  1.0977e+05               1.6715e+05                  2.2812e+05  2.30
min    1.0000e+06              1.0000e+00  -420.855000              0.000000                    -6827.310000  0.00
25%    1.0707e+06              1.0000e+00  0.000000                 45000.000000                0.000000  0.00
50%    1.0707e+06              2.8000e+01  0.000000                 112500.000000               0.000000  0.00
75%    1.0707e+06              4.8000e+01  89046.000000             180000.000000               0.000000  4.50
max    1.0707e+06              9.6000e+01  1055870.670000           1350000.000000              4.509079e+06  4.50
```

8 rows × 23 columns

Grouping features by type

```
# Grouping features by type
numerical_features = ['AMT_ID_PREV', 'NUM_INSTALMENT_NUMBER', 'AMT_BALANCE',
                      'AMT_CREDIT_LIMIT_ACTUAL', 'AMT_DRAWINGS_ATM_CURRENT',
                      'AMT_DRAWINGS_CURRENT', 'CNT_DRAWINGS_ATM_CURRENT',
                      'CNT_DRAWINGS_CURRENT', 'CNT_INSTALMENT_MATURED_BY_PREV',
                      'SK_DPD', 'SK_DPD_DEF', 'SK_DID_PREV', 'SK_DID_DEF']
categorical_features = ['NAME_CONTRACT_STATUS']

# Print the lists to verify
print("Numerical features:", numerical_features)
print("Categorical features:", categorical_features)
```

```
Numerical features: ['AMT_ID_PREV', 'NUM_INSTALMENT_NUMBER', 'AMT_BALANCE', 'AMT_CREDIT_LIMIT_ACTUAL', 'AMT_DRAWINGS_ATM_CURRENT', 'AMT_DRAWINGS_CURRENT', 'CNT_DRAWINGS_ATM_CURRENT', 'CNT_DRAWINGS_CURRENT', 'CNT_INSTALMENT_MATURED_BY_PREV', 'SK_DPD', 'SK_DPD_DEF', 'SK_DID_PREV', 'SK_DID_DEF']
Categorical features: ['NAME_CONTRACT_STATUS']
```

```
# Binning features for numerical features
for feature in numerical_features:
    bins = pd.cut(credit_card_balance[feature], bins=5)
    credit_card_balance[f'{feature}_BINNED'] = bins

# Print the first few rows to see the binned features
print("Binning complete. Sample of binned features:")
print(credit_card_balance[[f'{feature}_BINNED' for feature in numerical_features]].head())
```

```
Binning complete. Sample of binned features:
  AMT_ID_PREV_BINNED NUM_INSTALMENT_NUMBER_BINNED AMT_BALANCE_BINNED  \
0  (1000000.0, 1200...  (0.95, 19.2]                (-420.856, 84209.2]
1  (1000000.0, 1200...  (38.4, 57.6]                (-420.856, 84209.2]
2  (1000000.0, 1200...  (38.4, 57.6]                (-420.856, 84209.2]
3  (1000000.0, 1200...  (38.4, 57.6]                (-420.856, 84209.2]
4  (1000000.0, 1200...  (38.4, 57.6]                (-420.856, 84209.2]

  AMT_CREDIT_LIMIT_ACTUAL_BINNED AMT_DRAWINGS_ATM_CURRENT_BINNED  \
0                (0.999, 270000.]                (-6827.311, 1365.462]
1                (0.999, 270000.]                (-6827.311, 1365.462]
2                (0.999, 270000.]                (-6827.311, 1365.462]
3                (0.999, 270000.]                (-6827.311, 1365.462]
4                (0.999, 270000.]                (-6827.311, 1365.462]

  AMT_DRAWINGS_CURRENT_BINNED
0  (-2.245079e+06, 4.509079e...
1  (-2.245079e+06, 4.509079e...
2  (-2.245079e+06, 4.509079e...
3  (-2.245079e+06, 4.509079e...
4  (-2.245079e+06, 4.509079e...
```

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import PolynomialFeatures
from sklearn.preprocessing import Normalizer
from sklearn.preprocessing import Binarizer
from sklearn.preprocessing import KBinsDiscretizer
from sklearn.preprocessing import FunctionTransformer
from sklearn.preprocessing import QuantileTransformer
from sklearn.preprocessing import PowerTransformer
from sklearn.preprocessing import OrdinalEncoder
from sklearn.preprocessing import TargetEncoder
from sklearn.preprocessing import LabelBinarizer
from sklearn.preprocessing import MultiLabelBinarizer
from sklearn.preprocessing import SplineTransformer
from sklearn.preprocessing import KernelCenterer
from sklearn.preprocessing import StandardScaler, Normalizer, MinMaxScaler
from sklearn.pipeline import Pipeline, FeatureUnion
from sklearn.compose import ColumnTransformer
from sklearn.impute import SimpleImputer
from sklearn.feature_selection import SelectKBest, f_classif
from sklearn.decomposition import PCA
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.model_selection import cross_val_score
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.metrics import roc_curve, auc
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import RandomizedSearchCV
from sklearn.ensemble import VotingClassifier
from sklearn.ensemble import StackingClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import HistGradientBoostingClassifier
from sklearn.ensemble import IsolationForest
from sklearn.ensemble import LocalOutlierFactor
from sklearn.ensemble import OneClassSVM
from sklearn.ensemble import EllipticEnvelope
from sklearn.ensemble import MinCovDet
from sklearn.ensemble import LabelPropagation
from sklearn.ensemble import LabelSpreading
from sklearn.ensemble import SelfTrainingClassifier
from sklearn.ensemble import CalibratedClassifierCV
from sklearn.ensemble import FeatureAgglomeration
from sklearn.ensemble import GaussianProcessClassifier
from sklearn.ensemble import GaussianProcessRegressor
from sklearn.ensemble import KernelPCA
from sklearn.ensemble import IncrementalPCA
from sklearn.ensemble import TruncatedSVD
from sklearn.ensemble import NonNegativeMatrixFactorization
from sklearn.ensemble import FactorAnalysis
from sklearn.ensemble import LatentDirichletAllocation
from sklearn.ensemble import NMF
from sklearn.ensemble import MiniBatchNMF
from sklearn.ensemble import SparseCoder
from sklearn.ensemble import DictionaryLearning
from sklearn.ensemble import MiniBatchDictionaryLearning
from sklearn.ensemble import SparsePCA
from sklearn.ensemble import MiniBatchSparsePCA
from sklearn.ensemble import FastICA
from sklearn.ensemble import PCA
from sklearn.ensemble import KernelPCA
from sklearn.ensemble import IncrementalPCA
from sklearn.ensemble import TruncatedSVD
from sklearn.ensemble import NonNegativeMatrixFactorization
from sklearn.ensemble import FactorAnalysis
from sklearn.ensemble import LatentDirichletAllocation
from sklearn.ensemble import NMF
from sklearn.ensemble import MiniBatchNMF
from sklearn.ensemble import SparseCoder
from sklearn.ensemble import DictionaryLearning
from sklearn.ensemble import MiniBatchDictionaryLearning
from sklearn.ensemble import SparsePCA
from sklearn.ensemble import MiniBatchSparsePCA
from sklearn.ensemble import FastICA
```

Creating the full pipeline with data preparation and base classifier

1. Naive Bayes

```
from sklearn.naive_bayes import MultinomialNB

# Create pipeline
multinomial_nb = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('classifier', MultinomialNB())
])

# Train the model
multinomial_nb.fit(X_train, y_train)

# Predictions
y_pred = multinomial_nb.predict(X_test)
```

```
from sklearn.naive_bayes import GaussianNB

# Create pipeline
gaussian_nb = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('classifier', GaussianNB())
])

# Train the model
gaussian_nb.fit(X_train, y_train)

# Predictions
y_pred = gaussian_nb.predict(X_test)
```

Logging results

```
results['Multinomial Naive Bayes', 'accuracy', y_pred, y_test],
results['Multinomial Naive Bayes', 'precision', y_pred, y_test],
results['Multinomial Naive Bayes', 'recall', y_pred, y_test],
results['Multinomial Naive Bayes', 'f1', y_pred, y_test],
results['Multinomial Naive Bayes', 'roc_auc', y_pred, y_test],
results['Multinomial Naive Bayes', 'confusion_matrix', y_pred, y_test]

results['Gaussian Naive Bayes', 'accuracy', y_pred, y_test],
results['Gaussian Naive Bayes', 'precision', y_pred, y_test],
results['Gaussian Naive Bayes', 'recall', y_pred, y_test],
results['Gaussian Naive Bayes', 'f1', y_pred, y_test],
results['Gaussian Naive Bayes', 'roc_auc', y_pred, y_test],
results['Gaussian Naive Bayes', 'confusion_matrix', y_pred, y_test]
```

```
Using 0 features:
Accuracy: 0.7
Precision: 0.7
Recall: 0.7
F1: 0.7
ROC AUC: 0.7
Confusion Matrix:
[[7 3]
 [3 7]]

Using 1000 features:
Accuracy: 0.7
Precision: 0.7
Recall: 0.7
F1: 0.7
ROC AUC: 0.7
Confusion Matrix:
[[7 3]
 [3 7]]
```

```
# Performance Measures
results_pipeline_cross_validation_scores_1, y_pred, y_test],
results_pipeline_cross_validation_scores_2, y_pred, y_test],
results_pipeline_cross_validation_scores_3, y_pred, y_test],
results_pipeline_cross_validation_scores_4, y_pred, y_test],
results_pipeline_cross_validation_scores_5, y_pred, y_test],
results_pipeline_cross_validation_scores_6, y_pred, y_test]

results_pipeline_cross_validation_scores_1, y_pred, y_test],
results_pipeline_cross_validation_scores_2, y_pred, y_test],
results_pipeline_cross_validation_scores_3, y_pred, y_test],
results_pipeline_cross_validation_scores_4, y_pred, y_test],
results_pipeline_cross_validation_scores_5, y_pred, y_test],
results_pipeline_cross_validation_scores_6, y_pred, y_test]
```

Plotting ROC Curve

```
import matplotlib.pyplot as plt
from sklearn.metrics import roc_curve, auc
plot_roc_curve(multinomial_nb, X_test, y_test)
plot_roc_curve(gaussian_nb, X_test, y_test)
```

Confusion matrix for testing data

```
from sklearn.metrics import plot_confusion_matrix
plot_confusion_matrix(multinomial_nb, X_test, y_test)
```

Logistic Regression

```
from sklearn.linear_model import LogisticRegression

# Create pipeline
logistic = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('classifier', LogisticRegression())
])

# Train the model
logistic.fit(X_train, y_train)

# Predictions
y_pred = logistic.predict(X_test)
```

```
# Performance Measures
results_pipeline_cross_validation_scores_1, y_pred, y_test],
results_pipeline_cross_validation_scores_2, y_pred, y_test],
results_pipeline_cross_validation_scores_3, y_pred, y_test],
results_pipeline_cross_validation_scores_4, y_pred, y_test],
results_pipeline_cross_validation_scores_5, y_pred, y_test],
results_pipeline_cross_validation_scores_6, y_pred, y_test]
```

Plotting ROC Curve

```
import matplotlib.pyplot as plt
from sklearn.metrics import roc_curve, auc
plot_roc_curve(logistic, X_test, y_test)
```

Confusion matrix for testing data

```
from sklearn.metrics import plot_confusion_matrix
plot_confusion_matrix(logistic, X_test, y_test)
```

Result Analysis

Do we can see the results from some logistic regression performed better than Naive bayes. So we'll go ahead and submit the logistic regression model to our Kaggle leader or league.

Kaggle Submission

```
# Submission
submission_df = pd.DataFrame(y_pred, columns=['Survived'])
submission_df.to_csv('submission.csv', index=False)
print('Submission saved!')
```

Home
Leagues
Home
Competitions
Datasets
Models
Code
Discussions
Learn
More
Your Work
Recently Viewed
Titanic - Machine Learning from Disaster
Predict survival on the Titanic and get familiar with ML basics
Submit Predictions
Late Submission
Your Submissions
Submission Details
Submission Details
Submission Details

```
# 2.1.3
# 2.1.3
# 2.1.3
```

```
Training finished. The best score on validation, including TP/FP/FN:
[Parallel(n_jobs=-1)]: Using backend threadingcc_pipeline_1_logistic_cv_fold1, TrueNetworkcc_pipeline_1_logistic_cv_fold1, TrueNetworkcc_
[Parallel(n_jobs=-1)]: Done 1 out of 1 | elapsed: 0.0s remaining: 0.0s
[Parallel(n_jobs=-1)]: Done 1 out of 1 | elapsed: 0.0s finished
Fitting 1 folds for each of 1 candidates, totalling 1 fits
[Parallel(n_jobs=-1)]: Using backend threadingcc_pipeline_1_logistic_cv_fold1, TrueNetworkcc_pipeline_1_logistic_cv_fold1, TrueNetworkcc_
[Parallel(n_jobs=-1)]: Done 1 out of 1 | elapsed: 0.0s remaining: 0.0s
[Parallel(n_jobs=-1)]: Done 1 out of 1 | elapsed: 0.0s finished
```

```
[0 rows x 0 columns]
Training dataset shape: (1000, 20)
Testing dataset shape: (250, 20)
Training target shape: (1000,)
Testing target shape: (250,)
```