1. 处理空余的数字：我的处理方式是先将空格比例大于 10%的行个抛弃掉，然后对于剩下的空格，通过这一列不为空的格子计算出这个格子为空的概率，然后将这个概率的值替代这个空格。

```
df=pd.read_csv('fraudulent.csv')
null_ratios = df.isnull().mean(axis=1)
df = df[null_ratios <= 0.1] #丢弃空格率高的行
num=[]
rows,cols=df.shape
for i in range(cols):
    num.append(0)
for i in range(rows):
    for j in range(cols):
        if(df.iloc[i,j]!=0):
            num[j]+=1
for i in range(rows):
    for j in range(cols):
        if(df.iloc[i,j]!=0 and df.iloc[i,j]!=1):
            df.iloc[i,j]=num[j]/rows#用为 1 的概率替换这个空格
```

2. 按照 4:1 的比例划分训练集与测试集。

```
X = df.iloc[:, :-1]  # 所有行，除了最后一列的所有列作为特征
y = df.iloc[:, -1]   # 所有行，最后一列作为目标变量
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

3. 采用 k 临近算法训练

```
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
```

4. 根据训练结果对测试集进行预测，算出方差

```
y_pred = knn.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse}")
```

5.完整代码如下

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import PolynomialFeatures
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import mean_squared_error

df=pd.read_csv('fraudulent.csv')
null_ratios = df.isnull().mean(axis=1)
df = df[null_ratios <= 0.1]
```

```
num=[]
rows,cols=df.shape
for i in range(cols):
    num.append(0)
for i in range(rows):
    for j in range(cols):
        if(df.iloc[i,j]!=0):
            num[j]+=1
for i in range(rows):
    for j in range(cols):
        if(df.iloc[i,j]!=0 and df.iloc[i,j]!=1):
            df.iloc[i,j]=num[j]/rows

X = df.iloc[:, :-1]  # 所有行，除了最后一列的所有列作为特征
y = df.iloc[:, -1]   # 所有行，最后一列作为目标变量

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

knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)

y_pred = knn.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse}")
```