```
import pandas as pd
from sklearn.preprocessing import OneHotEncoder,LabelEncoder
from sklearn.preprocessing import StandardScaler,Imputer
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.naive_bayes import GaussianNB,MultinomialNB

df_train = pd.read_csv('/Users/User/Downloads/Dataset/train_new.csv',encoding = 'latin
1')

df_train = df_train.drop('building_id',axis=1)

#convert non numerical values to numerical values
label = LabelEncoder()

#scale all the values such that the mean is 0 and standard deviation is 1
scale = StandardScaler()
for col in df_train.columns.values:
    if df_train[col].dtypes == 'object':
        data = df_train[col].values
        df_train[col] = label.fit_transform(data)

y = df_train['damage_grade']

#impute missing values
df_columns = df_train.columns
impute = Imputer()
df_train[df_train.columns]= impute.fit_transform(df_train)

#print(df_train.isnull().sum())

#split data into x and y
df_train = df_train.drop('damage_grade',axis = 1)

x = df_train
#after scaling the values
x = scale.fit_transform(x)

encoder = OneHotEncoder()
encoder.fit_transform(df_train)

#train test split
train,test,label_train,label_test = train_test_split(x,y,train_size=0.80)


#andom forest
rf = RandomForestClassifier(n_estimators = 500)
rf.fit(train,label_train)
predcited = rf.predict(test)
print(accuracy_score(label_test,predcited))

#naive bayes
gbn = GaussianNB()
gbn.fit(train,label_train)
predicted = gbn.predict(test)
print(accuracy_score(label_test,predicted))

#knn
for i in range(2,26):
```

```
    knn = KNeighborsClassifier(n_neighbors=5)
    knn.fit(train,label_train)
    predicted = knn.predict(test)
    print(accuracy_score(label_test,predcited))

#plot an elbow graph to pick the value of k
```