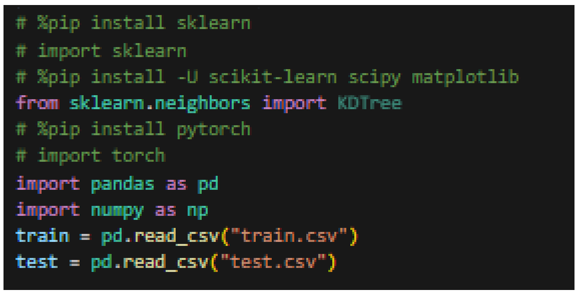
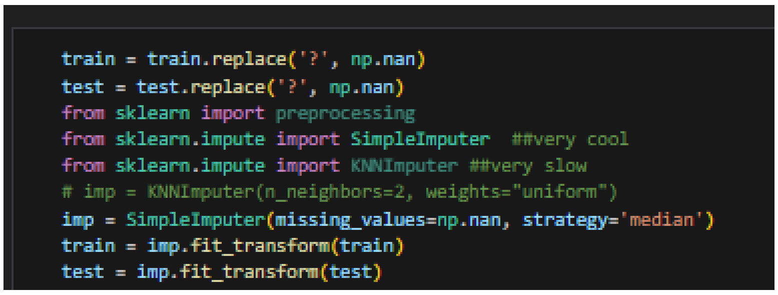
1. Библиотеки + Читаем дату



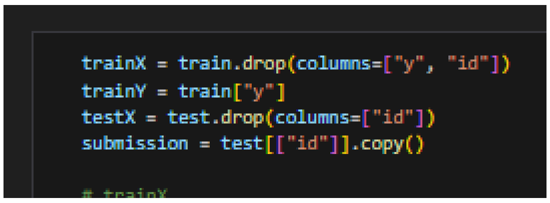
```
# %pip install sklearn
# import sklearn
# %pip install -U scikit-learn scipy matplotlib
from sklearn.neighbors import KDTree
# %pip install pytorch
# import torch
import pandas as pd
import numpy as np
train = pd.read_csv("train.csv")
test = pd.read_csv("test.csv")
```

2. Заполнять пропущенные



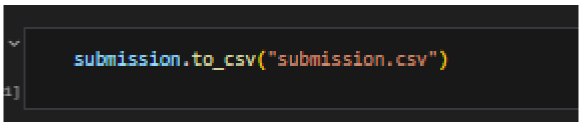
```
train = train.replace('?', np.nan)
test = test.replace('?', np.nan)
from sklearn import preprocessing
from sklearn.impute import SimpleImputer  ##very cool
from sklearn.impute import KNNImputer ##very slow
# imp = KNNImputer(n_neighbors=2, weights="uniform")
imp = SimpleImputer(missing_values=np.nan, strategy='median')
train = imp.fit_transform(train)
test = imp.fit_transform(test)
```

3. Расставляем приоритеты



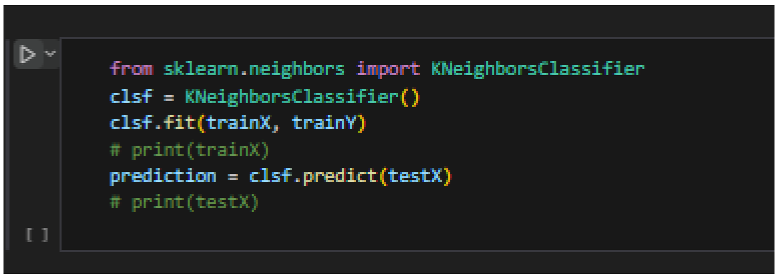
```
trainX = train.drop(columns=["y", "id"])
trainY = train["y"]
testX = test.drop(columns=["id"])
submission = test[["id"]].copy()

# trainX
```

4. Ответ



```
submission.to_csv("submission.csv")
```

5. Классификация Модели



```
from sklearn.neighbors import KNeighborsClassifier
clsf = KNeighborsClassifier()
clsf.fit(trainX, trainY)
# print(trainX)
prediction = clsf.predict(testX)
# print(testX)
```

```python
from sklearn.datasets import make_circles, make_classification, make_moons
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis
from sklearn.ensemble import AdaBoostClassifier, RandomForestClassifier
from sklearn.gaussian_process import GaussianProcessClassifier
from sklearn.gaussian_process.kernels import RBF
from sklearn.inspection import DecisionBoundaryDisplay
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier

names = [
    "Nearest Neighbors",
    "Linear SVM",
    "RBF SVM",
    "Gaussian Process",
    "Decision Tree",
    "Random Forest",
    "Neural Net",
    "AdaBoost",
    "Naive Bayes",
    "QDA",
]

classifiers = [
    KNeighborsClassifier(3),
    SVC(kernel="linear", C=0.025, random_state=42),
    SVC(gamma=2, C=1, random_state=42),
    GaussianProcessClassifier(1.0 * RBF(1.0), random_state=42),
    DecisionTreeClassifier(max_depth=5, random_state=42),
    RandomForestClassifier(
        max_depth=5, n_estimators=10, max_features=1, random_state=42
    ),
    MLPClassifier(alpha=1, max_iter=1000, random_state=42),
    AdaBoostClassifier(random_state=42),
    GaussianNB(),
    QuadraticDiscriminantAnalysis(),
]
```

Линейная регрессия модели

```python
from sklearn import linear_model
reg = linear_model.LinearRegression()
```

```python
>>> from sklearn import linear_model
>>> reg = linear_model.Ridge(alpha=.5)
```

```python
>>> from sklearn import linear_model
>>> reg = linear_model.LassoLars(alpha=.1)
```

```python
>>> from sklearn import linear_model
>>> X = [[0., 0.], [1., 1.], [2., 2.], [3., 3.]]
>>> Y = [0., 1., 2., 3.]
>>> reg = linear_model.BayesianRidge()
>>> reg.fit(X, Y)
```