|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ  Информатика и системы управления

КАФЕДРА Системы обработки информации и управления\_\_\_\_\_\_\_\_\_\_\_

Отчет по лабораторной работе № **4**

**«Подготовка обучающей и тестовой выборки, кросс-валидация и подбор гиперпараметров на примере метода ближайших соседей.»**

по дисциплине «Технологии машинного обучения»

Студент ИУ5-62Б  М.С. Вольвач

(Группа) (Подпись, дата) (И.О.Фамилия)

Преподаватель  Ю.Е. Гапанюк

(Подпись, дата) (И.О.Фамилия)

**Москва**

**2024**

Цель работы:

Изучение линейных моделей, SVM и деревьев решений.

Задание:

* Выберите набор данных (датасет) для решения задачи классификации или регрессии.
* В случае необходимости проведите удаление или заполнение пропусков и кодирование категориальных признаков.
* С использованием метода train\_test\_split разделите выборку на обучающую и тестовую.
* Обучите следующие модели:
  + одну из линейных моделей (линейную или полиномиальную регрессию при решении задачи регрессии, логистическую регрессию при решении задачи классификации);
  + SVM;
  + дерево решений.
* Оцените качество моделей с помощью двух подходящих для задачи метрик. Сравните качество полученных моделей.
* Постройте график, показывающий важность признаков в дереве решений.
* Визуализируйте дерево решений или выведите правила дерева решений в текстовом виде.

Описание набора данных:

Датасет содержит данные о кредитах на покупку электроники, которые были одобрены Tinkoff.ru.

Код программы:

import pandas as pd

import numpy as np

from matplotlib import pyplot as plt

import seaborn as sns

from sklearn.model\_selection import train\_test\_split, GridSearchCV, RandomizedSearchCV

from sklearn.neighbors import KNeighborsClassifier

from sklearn.preprocessing import MinMaxScaler, StandardScaler

from sklearn.linear\_model import LogisticRegression, LogisticRegressionCV

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score, roc\_auc\_score

from sklearn.tree import DecisionTreeClassifier, export\_graphviz, export\_text

from sklearn.svm import SVC

import graphviz

from IPython.display import Image

from warnings import simplefilter

simplefilter('ignore')

# записываем CSV-файл в объект DataFrame

data = pd.read\_csv('credit\_train\_preprocess.csv', encoding='cp1251', sep=',')

# смотрим на первые пять строк

data.head()

corr = data.corr().round(2)

f, ax = plt.subplots(figsize=(20, 20))

cmap = sns.diverging\_palette(120, 50, as\_cmap=True)

sns.heatmap(data=corr, cmap=cmap, annot=True, vmax=1.0, square=True, linewidths=.3, cbar\_kws={"shrink": .5}, ax=ax)#, vmax=1.0, square=True, xticklabels=2, yticklabels=2, linewidths=.3, cbar\_kws={"shrink": .5}, ax=ax)

plt.show()

corr = data[best\_params.index].corr().round(2)

f, ax = plt.subplots(figsize=(20, 20))

cmap = sns.diverging\_palette(120, 50, as\_cmap=True)

sns.heatmap(data=corr, cmap=cmap, annot=True, vmax=1.0, square=True, linewidths=.3, cbar\_kws={"shrink": .5}, ax=ax)#, vmax=1.0, square=True, xticklabels=2, yticklabels=2, linewidths=.3, cbar\_kws={"shrink": .5}, ax=ax)

plt.show()

plt.figure(figsize=(6, 8))

sns.heatmap(pd.DataFrame(data[np.append(best\_params.index.values, 'open\_account\_flg')].corr()['open\_account\_flg'].sort\_values(ascending=False)[1:]), vmin=-1, vmax=1, cmap=cmap, annot=True)

plt.show()

y = data['open\_account\_flg']

#X = data.drop('open\_account\_flg', axis=1)

X = data\_best

x\_train, x\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.75, random\_state=21)

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x\_train, y\_train, test\_size=0.3, random\_state=21)

scaler = MinMaxScaler().fit(x\_train)

x\_train = pd.DataFrame(scaler.transform(x\_train), columns=x\_train.columns)

x\_test = pd.DataFrame(scaler.transform(x\_test), columns=x\_train.columns)

x\_train.describe()

# обучаем логистическую регрессию

clf = LogisticRegression()

clf.fit(x\_train, y\_train);

plt.figure(figsize=(12, 4))

sns.barplot(data=importances.sort\_values(by='Важность', ascending=False), y='Признак', x='Важность', orient='h', )

plt.title('Важность признаков в дереве решений')

plt.show()

from sklearn import tree

fig, ax = plt.subplots(figsize=(30, 15))

clf = DecisionTreeClassifier(max\_depth = 5,

random\_state = 0)

clf.fit(x\_train, y\_train)

cn=['declined', 'accepted']

tree.plot\_tree(clf, fontsize=10, class\_names=cn, filled=True)

plt.show()

print('Логистическая регрессия')

print\_metrics(y\_test, y\_pred\_log)

print('\nМетод опорных векторов')

print\_metrics(y\_test, y\_pred\_svm)

print('\nДерево решений')

print\_metrics(y\_test, y\_pred\_tree)

Результат работы программы:







