**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра ИУ5.

Курс «Введение в анализ данных и машинное обучение»

Отчет по лабораторной работе №3

«Предсказание стоимости дома»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-42 |  | преподаватель каф. ИУ5 |
| Егор Хаметов |  | Гапанюк Ю.Е. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2018 г.

**Задание:**

Необходимо решить задачу предсказания стоимости дома в зависимости от его характеристик. Задача решается в рамках платформы онлайн-конкурсов по машинному обучению Kaggle.

**Текст программы:**

import pandas as pd

import matplotlib.pyplot as plt

import numpy as np

from scipy import stats

from sklearn.cross\_validation import train\_test\_split

from sklearn import linear\_model

from sklearn.metrics import mean\_absolute\_error, r2\_score

%matplotlib inline

data = pd.read\_csv('./KaggleLab3/train.csv')

plt.plot(data['GrLivArea'], data['SalePrice'], 'ro')

def cleaning(data):

categorical\_columns = [c for c in data.columns if data[c].dtype.name == 'object']

numerical\_columns = [c for c in data.columns if (data[c].dtype.name != 'object' and c != 'SalePrice')]

answer\_column = [c for c in data.columns if c == 'SalePrice']

#заполняем пустые количественные медианным значением

data = data.fillna(data.median(axis=0), axis=0)

#заполняем пустые категориальные самым частым значением по признаку

data\_describe = data.describe(include=[object])

for c in categorical\_columns:

data[c] = data[c].fillna(data\_describe[c]['top'])

#начинаем векторизацию - переводим категориальные признаки в количественные

binary\_columns = [c for c in categorical\_columns if data\_describe[c]['unique'] == 2] #бинарные

nonbinary\_columns = [c for c in categorical\_columns if data\_describe[c]['unique'] > 2] #небинарные

for c in binary\_columns:

top = data\_describe[c]['top']

top\_items = data[c] == top

data.loc[top\_items, c] = 0

data.loc[np.logical\_not(top\_items), c] = 1

data\_nonbinary = pd.get\_dummies(data[nonbinary\_columns])

#начинаем нормализацию количественных признаков

data\_numerical = data[numerical\_columns]

data\_numerical = (data\_numerical - data\_numerical.mean()) / data\_numerical.std()

data\_answer = data[answer\_column] #не требуется нормализация

#соединяем всё в таблицу

data = pd.concat((data\_numerical, data[binary\_columns], data\_nonbinary, data\_answer), axis=1)

data = pd.DataFrame(data, dtype=float)

return data

data = cleaning(data)

data.corr()

data.corr()['SalePrice'].abs().sort\_values(ascending=False)

data['GrLivArea'].corr(data['TotalBsmtSF'])

data = data[['OverallQual','GrLivArea','GarageCars','TotalBsmtSF','ExterQual\_TA','FullBath','BsmtQual\_Ex','TotRmsAbvGrd','YearBuilt','KitchenQual\_TA','GarageFinish\_Unf','KitchenQual\_Ex','SalePrice']]

plt.plot(data['GrLivArea'], data['SalePrice'], 'ro')

data=data[(np.abs(stats.zscore(data)) < 3).all(axis=1)]

plt.plot(data['GrLivArea'], data['SalePrice'], 'ro')

x = data.drop(('SalePrice'), axis=1) # входные фичи

y = data['SalePrice'] # ответ

x\_train, x\_valid, y\_train, y\_valid = train\_test\_split(x, y, test\_size = 0.5, random\_state = 11)

regr = linear\_model.LinearRegression(fit\_intercept=True)

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

y\_valid\_predict = regr.predict(x\_valid)

print('Coefficients: \n', regr.coef\_)

print("MAE: %.2f" % mean\_absolute\_error(y\_valid, y\_valid\_predict))

print('Variance score: %.2f' % r2\_score(y\_valid, y\_valid\_predict))

x\_test = pd.read\_csv('./KaggleLab3/test.csv')

y\_test = pd.read\_csv('./KaggleLab3/sample\_submission.csv')

x\_test = cleaning(x\_test)

x\_test = x\_test[['OverallQual','GrLivArea','GarageCars','TotalBsmtSF','ExterQual\_TA','FullBath','BsmtQual\_Ex','TotRmsAbvGrd','YearBuilt','KitchenQual\_TA','GarageFinish\_Unf','KitchenQual\_Ex']]

y\_test = y\_test[['SalePrice']]

y\_test\_predict = regr.predict(x\_test)

print('Coefficients: \n', regr.coef\_)

print("MAE: %.2f" % mean\_absolute\_error(y\_test, y\_test\_predict))

print('Variance score: %.2f' % r2\_score(y\_test, y\_test\_predict))

predictions = pd.DataFrame({'SalePrice':y\_test\_predict.round()}, index=range(1461,2920))

predictions.index.name = 'Id'

predictions.to\_csv('./KaggleLab3/predictions.csv')