|  |  |  |
| --- | --- | --- |
|  | |  |
| МИНОБРНАУКИ РОССИИ | |  |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **"МИРЭА - Российский технологический университет"**  **РТУ МИРЭА** | |  |
|  | Институт информационных технологий (ИТ) | |
|  | Кафедра прикладной математики | |

|  |  |
| --- | --- |
| **Отчет по практической работе №5.** | |
| **по дисциплине** | |
| **«Технологии и инструментарий анализа больших данных»** | |
|  | |
| Выполнил студент группы ИКБО- 22-20 | Никулин К.В. |
| Принял | Парамонов А.А. |

Москва 2023

Решение

1. Найти данные для классификации. Данные в группе повторяться не должны. Предобработать данные, если это необходимо.

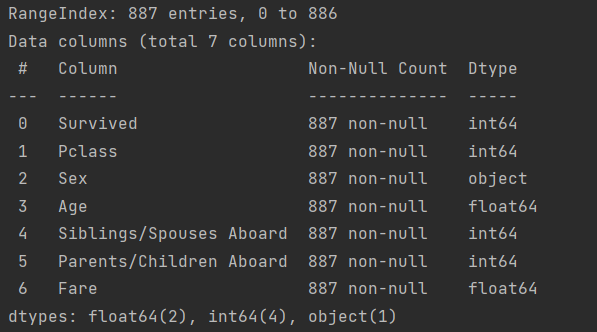


Рисунок 1 – Датасет пассажиры титаника

2. Изобразить гистограмму, которая показывает баланс классов. Сделать выводы.

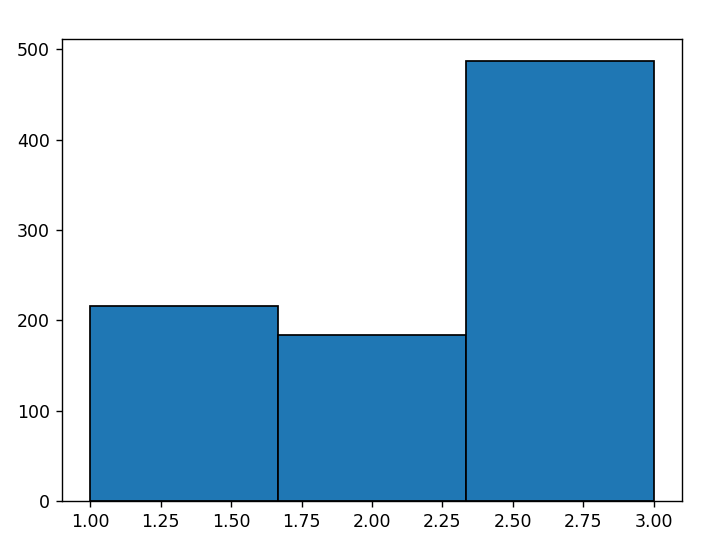


Рисунок 2 – Гистограммы классов

На графике видно, что данные достаточно сбалансированы.

3. Разбить выборку на тренировочную и тестовую. Тренировочная для обучения модели, тестовая для проверки ее качества.

Коэффициент train\_size=0.9.

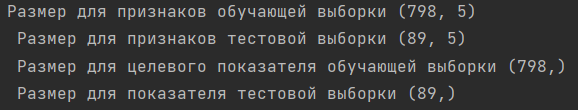


Рисунок 3 – Разбиение выборки

4. Применить алгоритмы классификации: логистическая регрессия, SVM, KNN. Построить матрицу ошибок по результатам работы моделей (использовать confusion\_matrix из sklearn.metrics).

Логистическая регрессия:

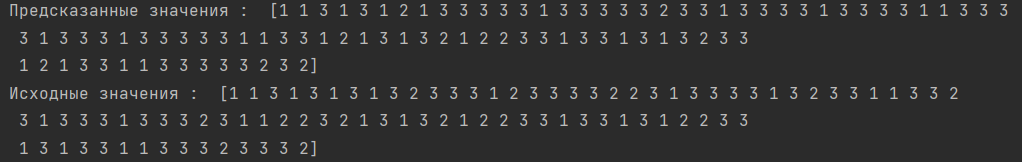


Рисунок 4 – Результат работы

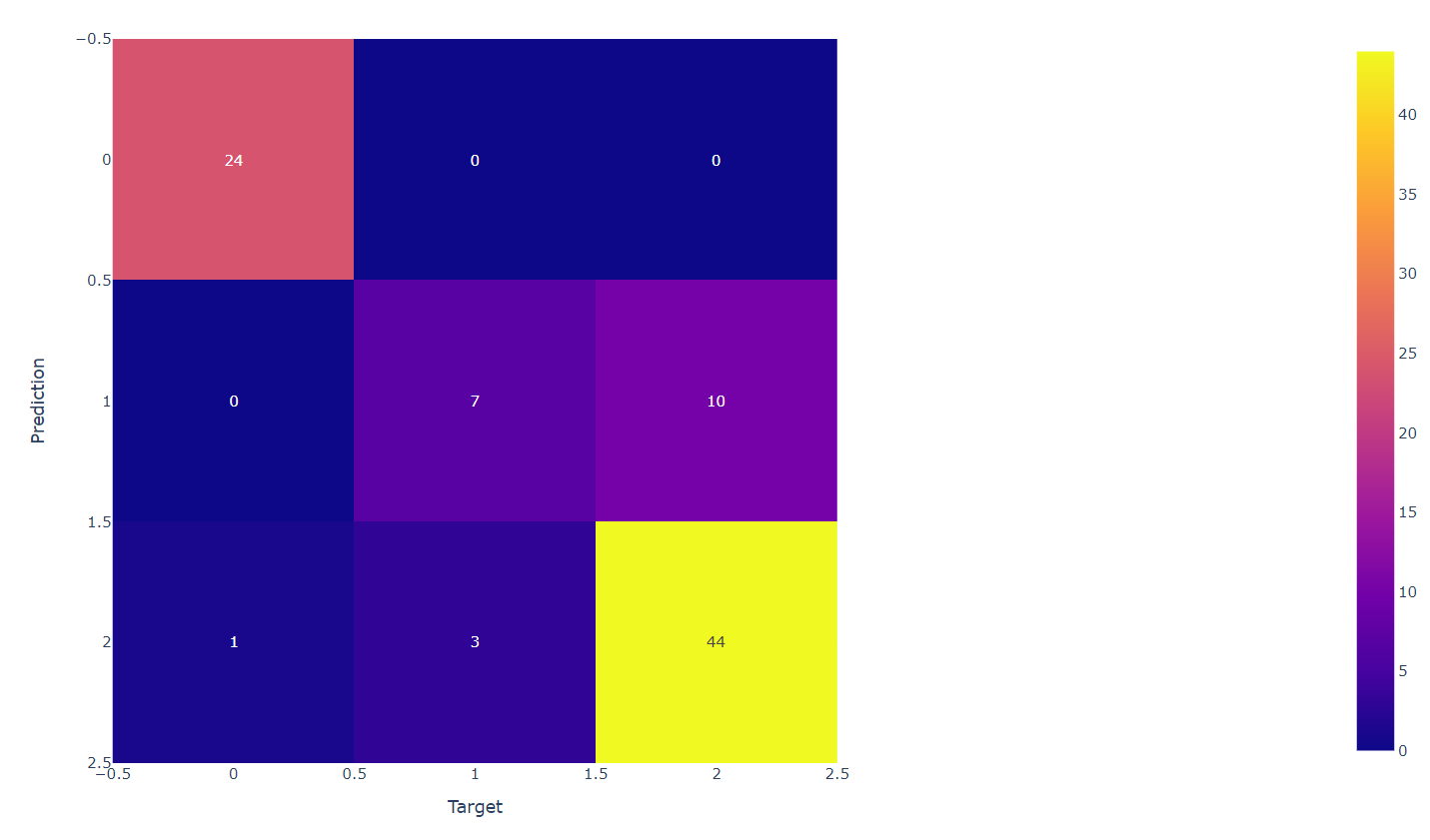


Рисунок 5 – Матрица ошибок логистической регрессии

SVM (метод опорных векторов):

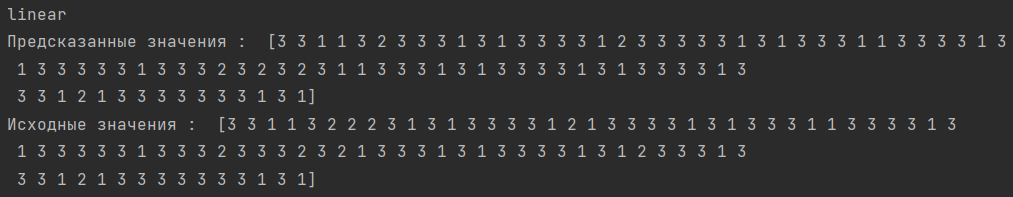


Рисунок 6 – Результат работы

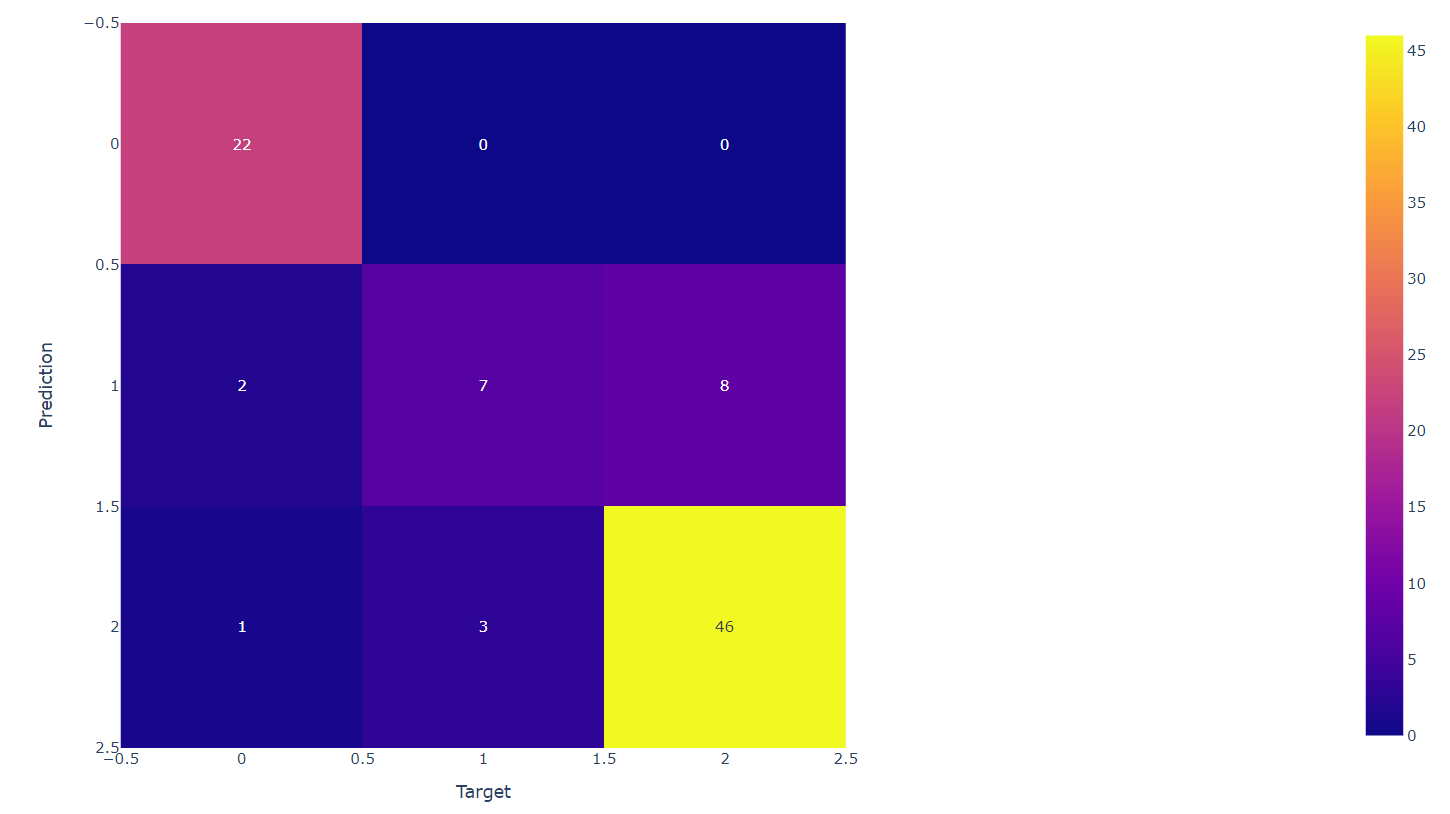


Рисунок 7 – Матрица ошибок SVM

KNN:

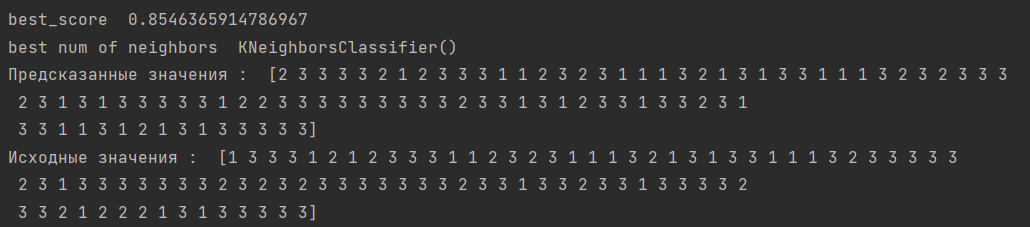


Рисунок 8 – Результат работы

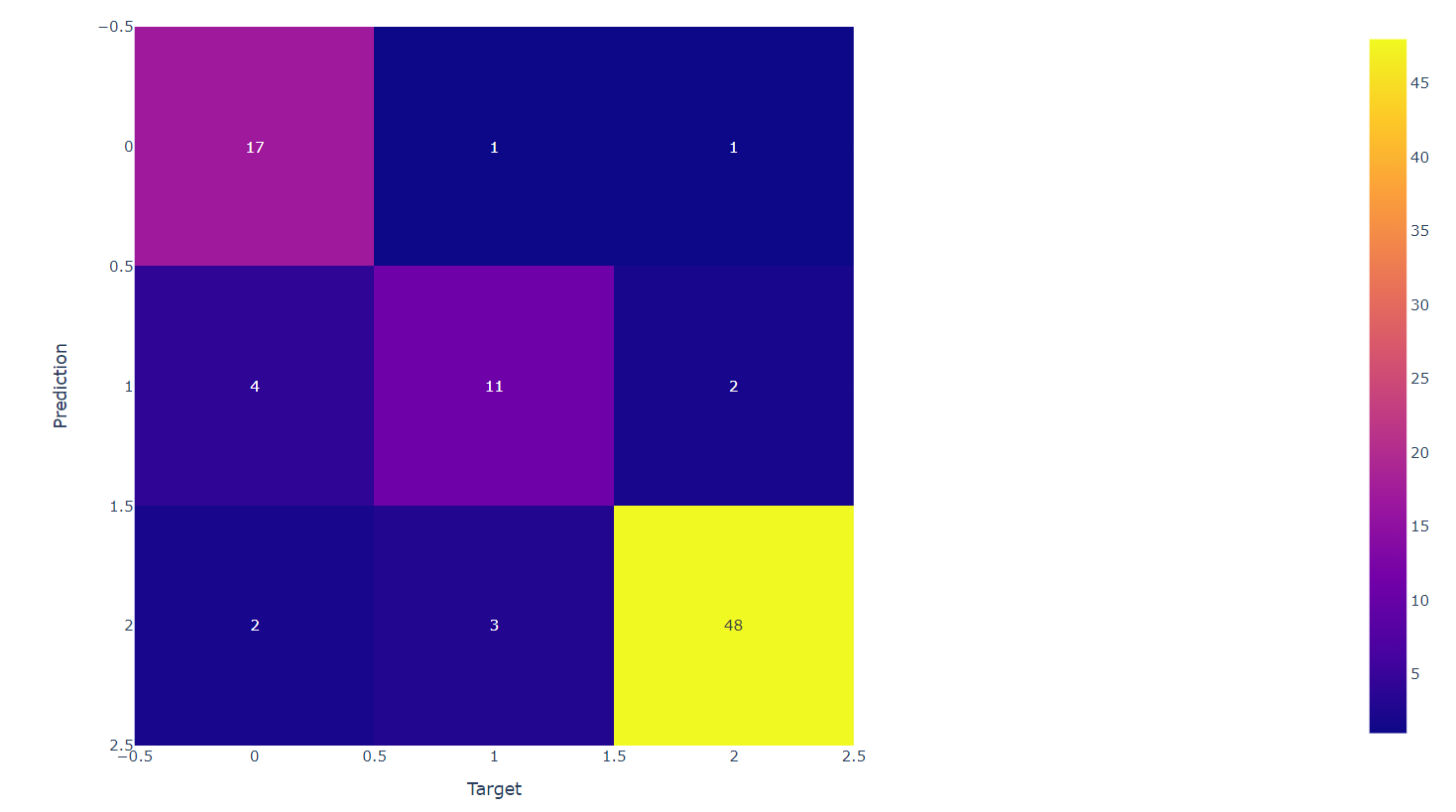


Рисунок 9 – Матрица ошибок KNN

5. Сравнить результаты классификации, используя accuracy, precision, recall и f1-меру (можно использовать classification\_report из sklearn.metrics). Сделать выводы.

Логистическая регрессия:

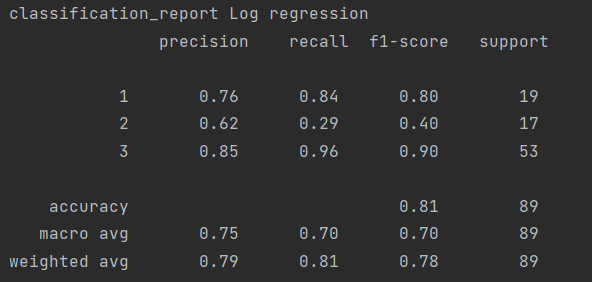


Рисунок 10 – Результат классификации логистической регрессии

SVM:

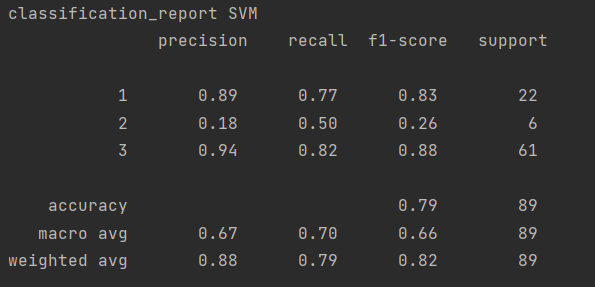


Рисунок 11 – Результат классификации SVM

KNN:

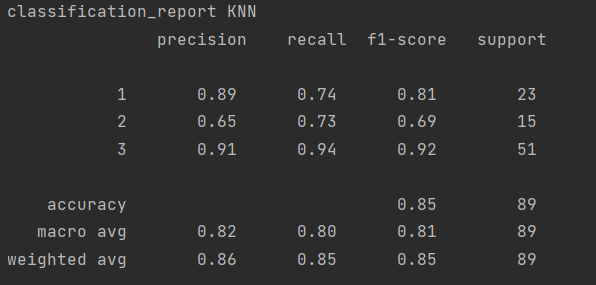


Рисунок 12 – Результат классификации KNN

Исходя из результатов, мы можем сделать вывод, что наименее точным алгоритмом классификации для данного датасета является SVM.

ПРИЛОЖЕНИЕ

Приложение 1- Код программы

import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
import plotly.express as px  
from sklearn.model\_selection import train\_test\_split, GridSearchCV  
from sklearn.linear\_model import LogisticRegression  
from sklearn.metrics import confusion\_matrix, classification\_report  
from sklearn.neighbors import KNeighborsClassifier  
from sklearn.svm import SVC  
  
  
def logreg(df):  
 predictors = df.drop('Pclass', axis=1)  
 target = df['Pclass']  
 x\_train, x\_test, y\_train, y\_test = train\_test\_split(predictors, target, train\_size=0.9, shuffle=True)  
 print('Размер для признаков обучающей выборки', x\_train.shape, '\n',  
 'Размер для признаков тестовой выборки', x\_test.shape, '\n',  
 'Размер для целевого показателя обучающей выборки', y\_train.shape, '\n',  
 'Размер для показателя тестовой выборки', y\_test.shape)  
 # Log regression  
 model = LogisticRegression()  
 model.fit(x\_train, y\_train)  
 y\_predict = model.predict(x\_test)  
 print('Предсказанные значения : ', y\_predict)  
 print('Исходные значения : ', np.array(y\_test))  
 print('classification\_report Log regression ', '\n',  
 classification\_report(y\_test, y\_predict))  
 plt.rcParams['figure.figsize'] = (10, 10)  
 fig = px.imshow(confusion\_matrix(y\_test, y\_predict), text\_auto=True)  
 fig.update\_layout(xaxis\_title='Target', yaxis\_title='Prediction')  
 fig.show()  
 # SVM  
 param\_kernel = ('linear', 'rbf', 'poly', 'sigmoid')  
 parameters = {'kernel': param\_kernel}  
 model\_svc = SVC()  
 grid\_search\_svm = GridSearchCV(estimator=model\_svc, param\_grid=parameters, cv=6)  
 grid\_search\_svm.fit(x\_train, y\_train)  
 best\_model\_svc = grid\_search\_svm.best\_estimator\_  
 print(best\_model\_svc.kernel)  
 svm\_preds = best\_model\_svc.predict(x\_test)  
 print('Предсказанные значения : ', svm\_preds)  
 print('Исходные значения : ', np.array(y\_test))  
 print('classification\_report SVM ', '\n',  
 classification\_report(svm\_preds, y\_test))  
 fig = px.imshow(confusion\_matrix(y\_test, svm\_preds), text\_auto=True)  
 fig.update\_layout(xaxis\_title='Target', yaxis\_title='Prediction')  
 fig.show()  
 # KNN  
 number\_of\_neighbors = np.arange(3, 10)  
 model\_knn = KNeighborsClassifier()  
 params = {'n\_neighbors': number\_of\_neighbors}  
 grid\_search = GridSearchCV(estimator=model\_knn, param\_grid=params, cv=6)  
 grid\_search.fit(x\_train, y\_train)  
 print("best\_score ", grid\_search.best\_score\_)  
 print("best num of neighbors ", grid\_search.best\_estimator\_)  
 knn\_preds = grid\_search.predict(x\_test)

Продолжение – Приложение 1

print('Предсказанные значения : ', knn\_preds)  
 print('Исходные значения : ', np.array(y\_test))  
 print('classification\_report KNN ', '\n',  
 classification\_report(knn\_preds, y\_test))  
 fig = px.imshow(confusion\_matrix(y\_test, knn\_preds), text\_auto=True)  
 fig.update\_layout(xaxis\_title='Target', yaxis\_title='Prediction')  
 fig.show()  
 return  
  
  
def main():  
 df = pd.read\_csv("Titanic.csv")  
 df.drop('Sex', axis=1, inplace=True)  
 print(df.info())  
 print(df.head())  
 plt.hist(df['Pclass'], bins=3, ec='black')  
 plt.show()  
 logreg(df)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()