Министерство науки и высшего образования Российской Федерации Федеральное государственноебюджетное 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 ИУ5 «Системы обработки информации и управления»

Курс «Технологии машинного обучения»

Отчёт по рубежному контролю №2

«Методы построения моделей машинного обучения»Вариант №16

Выполнил:

студент группы ИУ5-62Б

Поддубный М.Н.

Преподаватель:

Гапанюк Ю. Е.

2023 г.

Задание. Для заданного набора данных – Heart Disease Dataset постройте модели классификации или регрессии (в зависимости от конкретной задачи, рассматриваемой в наборе данных). Для построения моделей используйте: Метод опорных векторов и случайный лес. Оцените качество моделей на основе подходящих метрик качества (не менее двух метрик).

Выполнение работы

Импортируем нужные библиотеки и загружаем датасет

In [114]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn
from sklearn.ensemble import RandomForestRegressor
from sklearn import preprocessing, linear_model
from sklearn.neighbors import KNeighborsClassifier, KNeighborsRegressor
from sklearn.model_selection import train_test_split
from sklearn import neighbors
from sklearn.metrics import mean_squared_error
from math import sqrt
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.metrics import r2_score
%matplotlib inline
sns.set(style="ticks")
from sklearn import tree
from sklearn.svm import SVR, NuSVR, LinearSVR
from sklearn.preprocessing import LabelEncoder, StandardScaler
```

In [115]:
```
df = pd.read_csv('restaurant-scores-lives-standard.csv')
```

Проверяем на наличие пропусков и заполняем их

:
```
df.isnull().sum()
```

:
```
business_id                      0
business_name                    0
business_address                 0
business_city                    0
business_state                   0
business_postal_code          1017
business_latitude            19555
business_longitude           19555
business_location            19555
business_phone_number        36938
inspection_id                    0
inspection_date                  0
inspection_score             13609
inspection_type                  0
violation_id                 12869
violation_description        12869
risk_category                12869
Neighborhoods (old)          19593
Police Districts             19593
Supervisor Districts         19593
Fire Prevention Districts    19645
Zip Codes                    19575
Analysis Neighborhoods       19593
dtype: int64
```

In [119]:
```
df['inspection_score'].median()
```

Out[119]:
```
87.0
```

In [120]:
```
df['inspection_score'].fillna(df['inspection_score'].median(), inplace=True)
```

Удаляем признаки класса object

In [121]:
```
df.drop(['business_name'], axis=1, inplace=True)
df.drop(['business_postal_code'],axis=1, inplace=True)
df.drop(['inspection_type'],axis=1, inplace=True)
df.drop(['violation_id'],axis=1, inplace=True)
df.drop(['violation_description'],axis=1, inplace=True)
df.drop(['inspection_date'],axis=1, inplace=True)
df.drop(['risk_category'],axis=1, inplace=True)
df.drop(['business_address'],axis=1, inplace=True)
df.drop(['business_city'],axis=1, inplace=True)
df.drop(['business_state'],axis=1, inplace=True)
df.drop(['business_location'],axis=1, inplace=True)
df.drop(['inspection_id'],axis=1, inplace=True)
```

In [122]:
```
df.info()
```

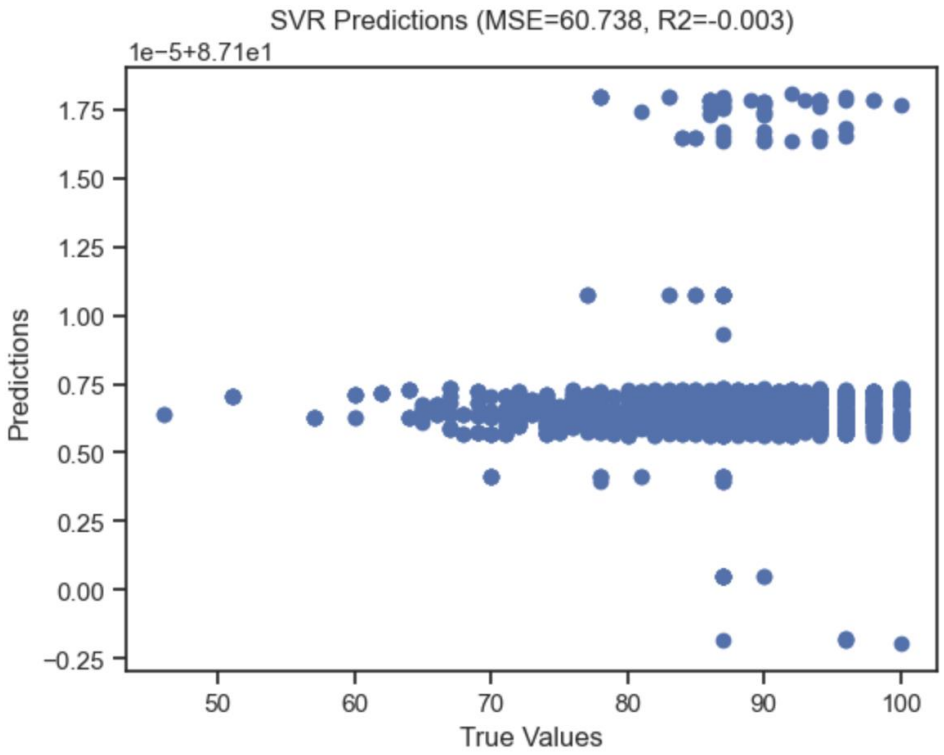
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 53972 entries, 0 to 53971
Data columns (total 11 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   business_id               53972 non-null  int64
 1   business_latitude         34417 non-null  float64
 2   business_longitude        34417 non-null  float64
 3   business_phone_number     17034 non-null  float64
 4   inspection_score          53972 non-null  float64
 5   Neighborhoods (old)       34379 non-null  float64
 6   Police Districts          34379 non-null  float64
 7   Supervisor Districts      34379 non-null  float64
 8   Fire Prevention Districts 34327 non-null  float64
 9   Zip Codes                 34397 non-null  float64
 10  Analysis Neighborhoods    34379 non-null  float64
dtypes: float64(10), int64(1)
memory usage: 4.5 MB
```

Убираем нулевые строки

In [123]:
```
df= df.dropna()
```

Метод опороных вектров

In [125]:
```
# Разделяем данные на тренировочный и тестовый наборы
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Обучаем модель
svr = SVR(kernel='rbf')
svr.fit(X_train, y_train)

# Получаем предсказания на тестовом наборе
y_pred = svr.predict(X_test)

# Оцениваем качество модели
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Строим график
plt.scatter(y_test, y_pred)
plt.xlabel('True Values')
plt.ylabel('Predictions')
plt.title('SVR Predictions (MSE={:.3f}, R2={:.3f})'.format(mse, r2))
plt.show()
```

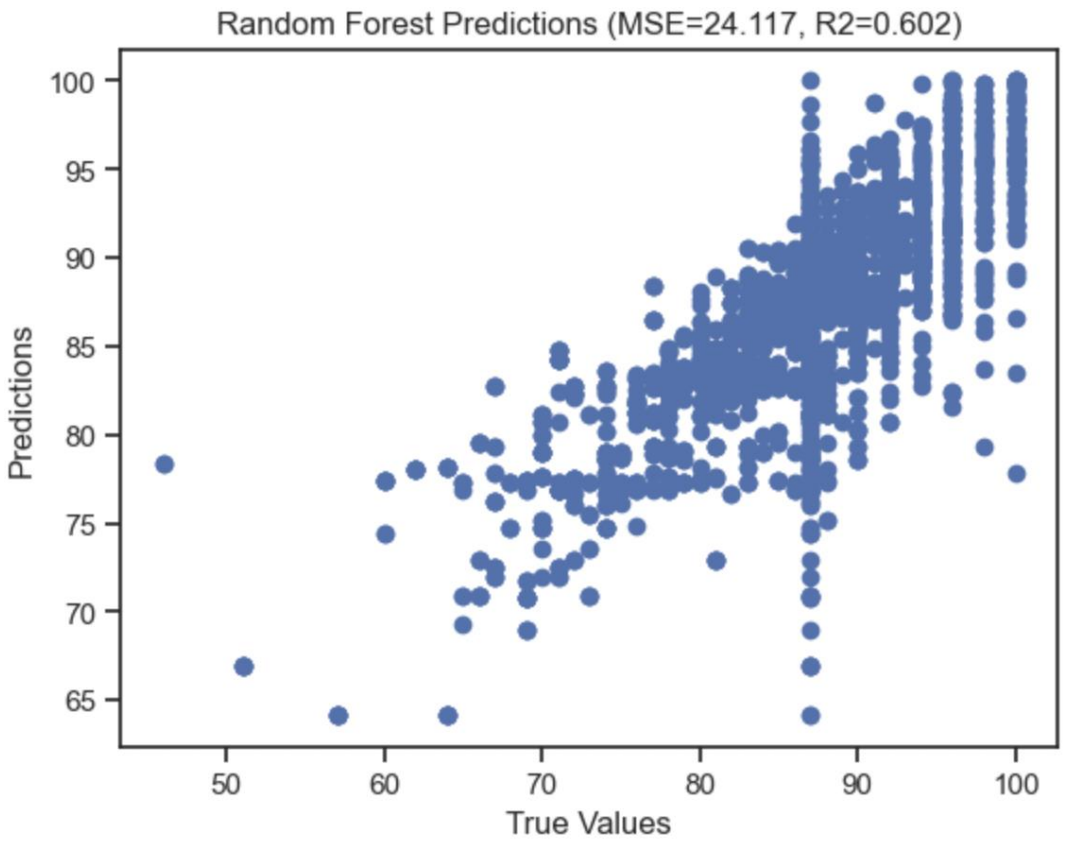


Случайный лес

In [126]:
```
# Замена пропущенных значений и разделение данных на тренировочный и тестовый наборы
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Обучение модели случайного леса
rf = RandomForestRegressor(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)

# Получение предсказаний на тестовом наборе
y_pred = rf.predict(X_test)

# Оценка качества модели
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print('Random Forest MSE:', mse)
print('Random Forest R2:', r2)
```

```
Random Forest MSE: 24.11714339840594
Random Forest R2: 0.6016206975660595
```

In [127]:
```
plt.scatter(y_test, y_pred)
plt.xlabel('True Values')
plt.ylabel('Predictions')
plt.title('Random Forest Predictions (MSE={:.3f}, R2={:.3f})'.format(mse, r2))
plt.show()
```



Вывод:

MSE измеряет среднеквадратичную ошибку предсказания модели, а R2 показывает, насколько хорошо модель соответствует данным и может быть интерпретирована как доля объясненной дисперсии в данных. Чем ближе значение MSE к нулю и R2 к единице, тем лучше качество модели.

В моей работе эти данные далеки от идеала, значит модели случайного леса и метод опорных векторов не подходят для этой модели.

Нужно еще раз проанализировать данные и параметры для более точной оценки модели.