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 ИУ5 «Системы обработки информации и управления»

Отчёт по рубежному контролю №2

«Технологии машинного обучения»

Вариант 18

Выполнила:

студентка группы ИУ5-63Б

Шаповалова В.В.

Преподаватель:

Гапанюк Ю. Е.

2023 г.

Задание:

Для заданного набора данных (по Вашему варианту) постройте модели классификации или регрессии (в зависимости от конкретной задачи, рассматриваемой в наборе данных). Для построения моделей используйте методы 1 и 2 (по варианту для Вашей группы). Оцените качество моделей на основе подходящих метрик качества (не менее двух метрик). Для построения моделей необходимо выполнить требуемую предобработку данных: заполнение пропусков, кодирование категориальных признаков, и т.д.

Группа	**Метод №1**	**Метод №2**
ИУ5-63Б, ИУ5Ц-83Б	Дерево решений	Случайный лес

https://www.kaggle.com/datasets/rhuebner/human-resources-data-set

Решение:

Загружаем датасет и подключаем необходимые библиотеки:

```
[166]: import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeRegressor, export_graphviz
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_absolute_error, mean_squared_error
from sklearn.metrics import median_absolute_error, r2_score
%matplotlib inline
```

```
[167]: df = pd.read_csv('HRDataset_v14.csv')
df.head()
```

Out[167]:

	Employee_Name	EmpID	MarriedID	MaritalStatusID	GenderID	EmpStatusID	DeptID	PerfScoreID	FromDiversityJobFairID	Salary	...	ManagerName	Manag
0	Adinolfi, Wilson K	10026	0	0	1	1	5	4	0	62506	...	Michael Albert	
1	Ait Sidi, Karthikeyan	10084	1	1	1	5	3	3	0	104437	...	Simon Roup	
2	Akinkuolie, Sarah	10196	1	1	0	5	5	3	0	64955	...	Kissy Sullivan	
3	Alagbe,Trina	10088	1	1	0	1	5	3	0	64991	...	Elijah Gray	
4	Anderson, Carol	10069	0	2	0	5	5	3	0	50825	...	Webster Butler	

5 rows × 36 columns

```
In [168]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 311 entries, 0 to 310
Data columns (total 36 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Employee_Name               311 non-null    object
 1   EmpID                       311 non-null    int64
 2   MarriedID                   311 non-null    int64
 3   MaritalStatusID             311 non-null    int64
 4   GenderID                    311 non-null    int64
 5   EmpStatusID                 311 non-null    int64
 6   DeptID                      311 non-null    int64
 7   PerfScoreID                 311 non-null    int64
 8   FromDiversityJobFairID      311 non-null    int64
 9   Salary                      311 non-null    int64
 10  Termd                       311 non-null    int64
 11  PositionID                  311 non-null    int64
 12  Position                    311 non-null    object
 13  State                       311 non-null    object
 14  Zip                         311 non-null    int64
 15  DOB                         311 non-null    object
 16  Sex                         311 non-null    object
 17  MaritalDesc                 311 non-null    object
 18  CitizenDesc                 311 non-null    object
 19  HispanicLatino              311 non-null    object
 20  RaceDesc                    311 non-null    object
 21  DateofHire                  311 non-null    object
 22  DateofTermination           104 non-null    object
 23  TermReason                  311 non-null    object
 24  EmploymentStatus            311 non-null    object
```

Посчитаем количество пустых значений:

```
In [169]: df.isna().sum()

Out[169]: Employee_Name                0
          EmpID                        0
          MarriedID                    0
          MaritalStatusID              0
          GenderID                     0
          EmpStatusID                  0
          DeptID                       0
          PerfScoreID                  0
          FromDiversityJobFairID       0
          Salary                       0
          Termd                        0
          PositionID                   0
          Position                     0
          State                        0
          Zip                          0
          DOB                          0
          Sex                          0
          MaritalDesc                  0
          CitizenDesc                  0
          HispanicLatino               0
          RaceDesc                     0
          DateofHire                   0
          DateofTermination          207
          TermReason                   0
          EmploymentStatus             0
          Department                   0
          ManagerName                  0
          ManagerID                    8
          RecruitmentSource            0
          PerformanceScore             0
          EngagementSurvey             0
          EmpSatisfaction              0
          SpecialProjectsCount         0
          LastPerformanceReview_Date   0
          DaysLateLast30               0
          Absences                     0
          dtype: int64
```

Поскольку пропуски находятся в малоинформативных столбцах, удалим их, а также удалим неинформативные столбцы:

```
In [170]: df = df.dropna()

In [171]: df = df.drop(['Position', 'EmploymentStatus', 'Sex', 'Department', 'ManagerName', 'PerformanceScore', 'Employee_Name', 'MaritalDe
```

Кодирование категориальных признаков:

```
In [172]: df["CitizenDesc"] = df["CitizenDesc"].astype('category')
          df["State"] = df["State"].astype('category')
          df["HispanicLatino"] = df["HispanicLatino"].astype('category')
          df["RaceDesc"] = df["RaceDesc"].astype('category')
          df["RecruitmentSource"] = df["RecruitmentSource"].astype('category')
          #Назначить закодированную переменную новосу столбцу с помощью метода доступа
          df["CitizenDesc_cat"] = df["CitizenDesc"].cat.codes
          df["State_cat"] = df["State"].cat.codes
          df["HispanicLatino_cat"] = df["HispanicLatino"].cat.codes
          df["RecruitmentSource_cat"] = df["RecruitmentSource"].cat.codes

In [173]: df = df.drop(['CitizenDesc','State','HispanicLatino','RaceDesc', 'RecruitmentSource'], axis = 1)
```

Мы преобразуем категориальные признаки в числовые значения, используя метод кодирования категорий. Каждое уникальное значение категориального признака заменяется числом, начиная с 0.

Разделение данных на обучающую и тестовую выборки:

```
In [200]: y = df['Salary']
          X = df.drop('Salary', axis=1)
          X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

Мы разделяем данные на обучающую и тестовую выборки в соотношении 80%/20% с использованием функции train_test_split. X представляет набор признаков, а y – целевую переменную.

Обучение моделей и оценка качества:

```
In [205]: def test_model(model):
              print("mean_absolute_error:",
                    mean_absolute_error(y_test, model.predict(X_test)))
              print("mean_squared_error:",
                    mean_squared_error(y_test, model.predict(X_test)))
```

```
In [206]: dt_none = DecisionTreeRegressor(random_state=0)
          dt_none.fit(X_train, y_train)
          test_model(dt_none)
```

```
mean_absolute_error: 11471.333333333334
mean_squared_error: 247339659.42857143
```

```
In [208]: rf_model = RandomForestRegressor(random_state=0)
          rf_model.fit(X_train, y_train)
          test_model(rf_model)
```

```
mean_absolute_error: 11167.573809523808
mean_squared_error: 220309428.82127145
```

Создаются две модели - модель дерева решений и модель случайного леса. Каждая модель обучается на обучающих данных (X_train и y_train), а затем используется для получения прогнозов на тестовых данных (X_test).

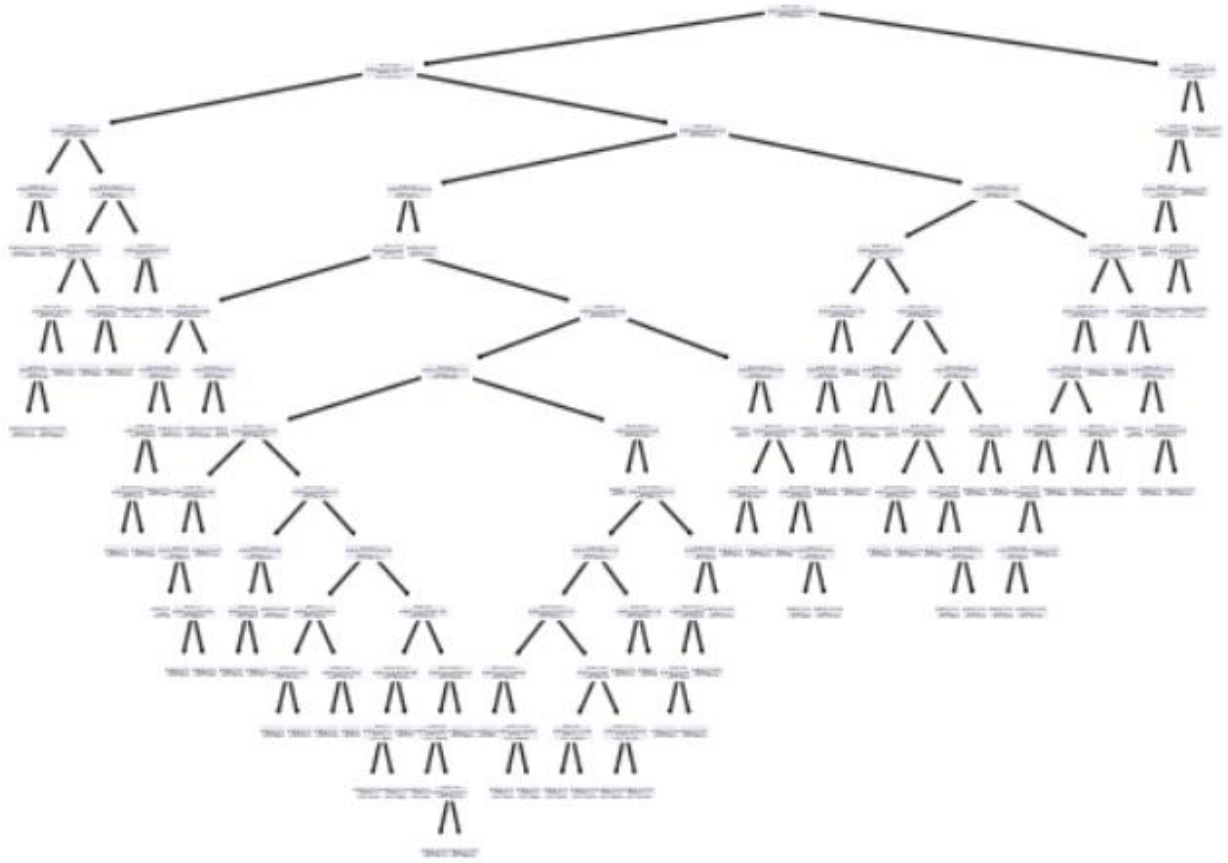
Для оценки качества прогнозов используются три метрики ошибки – mean absolute error и mean squared error. Для каждой модели рассчитываются значения этих метрик на тестовых данных.

Выводятся значения метрик для каждой модели. Интерпретация этих метрик зависит от контекста и задачи, которую решает модель. Обычно их меньшие значения свидетельствуют о лучшей точности прогнозов.

Визуализация дерева решений:

Мы визуализируем дерево решений с помощью функции plot_tree из библиотеки scikit-learn.

```
In [203]: tree.plot_tree(dt_none)
```



Вывод:

В данном примере были использованы две метрики качества - средняя абсолютная ошибка и средняя квадратичная ошибка.

Mean absolute error - это средняя абсолютная разница между прогнозами и фактическими значениями. Она показывает, насколько сильно отклоняются предсказания модели от реальных значений. Метрика является более понятной метрикой, так как ее значения можно интерпретировать как фактические расстояния между прогнозами и реальными значениями.

Mean squared error - это средняя квадратичная ошибка между предсказаниями и фактическими значениями. Она показывает, насколько сильно отличаются прогнозы модели от реальных значений. Метрика чувствительнее к большим ошибкам, так как она возводит разницу в квадрат.

Оценка качества моделей на тестовых данных показала, что модель случайного леса показывает лучшие показатели в сравнении с деревом решений. Однако для окончательных выводов необходима более обширная оценка, прежде всего, на больших объемах данных, чтобы подтвердить или опровергнуть эти выводы.