Сергеев М.Ю. ИУ5-63Б

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). Какие метрики качества Вы использовали и почему? Какие выводы Вы можете сделать о качестве построенных моделей? Для построения моделей необходимо выполнить требуемую предобработку данных: заполнение пропусков, кодирование категориальных признаков, и т.д.

In [1]:

```
import pandas as pd
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
from sklearn.preprocessing import LabelEncoder
from io import StringIO
from sklearn.tree import export_graphviz
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import MinMaxScaler
from IPython.display import Image
import matplotlib.pyplot as plt
import numpy as np
```

In [2]:

```
# Загрузка и первичный анализ данных
df = pd.read_csv('C:/Users/maxim/OneDrive/Рабочий стол/ТМО/HRDataset_v14.csv', sep=',')
```

In [3]:

```
# размер набора данных
df.shape
```

Out[3]:

```
(311, 36)
```

In [4]:

```
# типы колонок
df.dtypes
```

Out[4]:

```
Employee_Name              object
EmpID                       int64
MarriedID                   int64
MaritalStatusID             int64
GenderID                    int64
EmpStatusID                 int64
DeptID                      int64
PerfScoreID                 int64
FromDiversityJobFairID      int64
Salary                      int64
Termd                       int64
PositionID                  int64
Position                   object
State                      object
Zip                         int64
DOB                        object
Sex                        object
MaritalDesc                object
CitizenDesc                object
HispanicLatino             object
RaceDesc                   object
DateofHire                 object
DateofTermination          object
TermReason                 object
EmploymentStatus           object
Department                 object
ManagerName                object
ManagerID                 float64
RecruitmentSource          object
PerformanceScore           object
EngagementSurvey          float64
EmpSatisfaction             int64
SpecialProjectsCount        int64
LastPerformanceReview_Date    object
DaysLateLast30              int64
Absences                    int64
dtype: object
```

In [5]:

```
# проверим есть ли пропущенные значения
```

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

Out[5]:

```
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
```

In [6]:

```
# Первые 5 строк датасета
df.head(100)
```

Out[6]:

	Employee_Name	EmpID	MarriedID	MaritalStatusID	GenderID	EmpStatusID	DeptID	PerfScoreID	FromDiversityJobFairID	Salary	...	ManagerName	Mar
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	...	Elijiah Gray	
4	Anderson, Carol	10069	0	2	0	5	5	3	0	50825	...	Webster Butler	
...	...	...	...	...	...	...	...	...	...	...	...	...	
95	Forrest, Alex	10305	1	1	1	1	6	3	0	70187	...	Lynn Daneault	
96	Foss, Jason	10015	0	0	1	1	3	4	0	178000	...	Jennifer Zamora	
97	Foster-Baker, Amy	10080	1	1	0	1	1	3	0	99351	...	Board of Directors	
98	Fraval, Maruk	10258	0	0	1	1	6	3	0	67251	...	Lynn Daneault	
99	Galia, Lisa	10273	0	0	0	1	3	3	0	65707	...	Eric Dougall	

100 rows × 36 columns

In [7]:

```
df=df.drop(columns=['Employee_Name', 'Position','State','DOB','Sex','MaritalDesc','CitizenDesc','HispanicLatino','RaceDesc','DateofHire','DateofTermination',
```

In [8]:

```
df=df.dropna()
```

In [9]:

```
df_X_train, df_X_test, df_y_train, df_y_test = train_test_split(
    df.drop(columns='Salary'), df['Salary'], test_size=0.2, random_state=171)
```

Дерево решений

In [10]:

```
tree = DecisionTreeRegressor()
tree.fit(df_X_train, df_y_train)
```

Out[10]:

```
DecisionTreeRegressor()
```

In [11]:

```
tree_predict = tree.predict(df_X_test)
```

Случаный лес

In [12]:

```
forest = RandomForestRegressor()
forest.fit(df_X_train, df_y_train)
```

Out[12]:

```
RandomForestRegressor()
```

In [13]:

```
forest_predict = forest.predict(df_X_test)
```

Оценка моделей

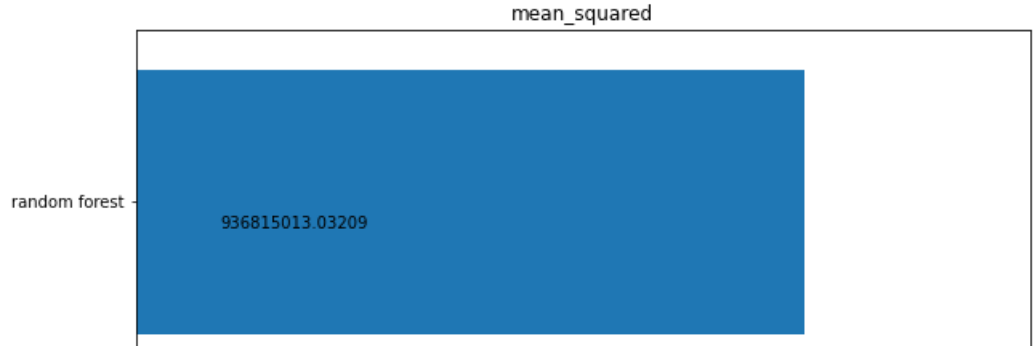
Для оценки будем использовать три метрики: **Средняя квадратичная ошибка**, **Средняя абсолютная ошибка**, **R2 score**.

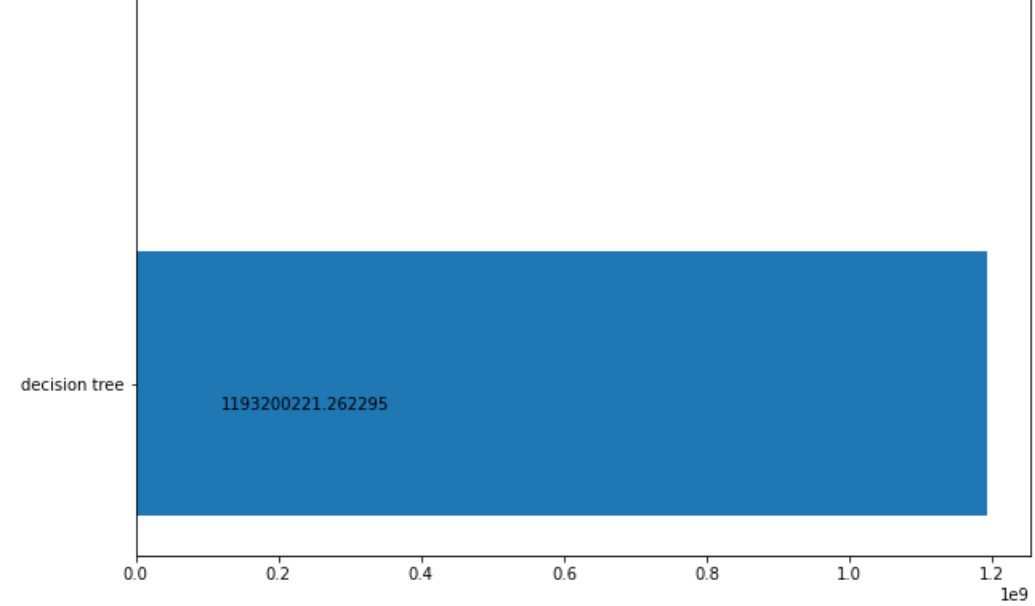
In [14]:

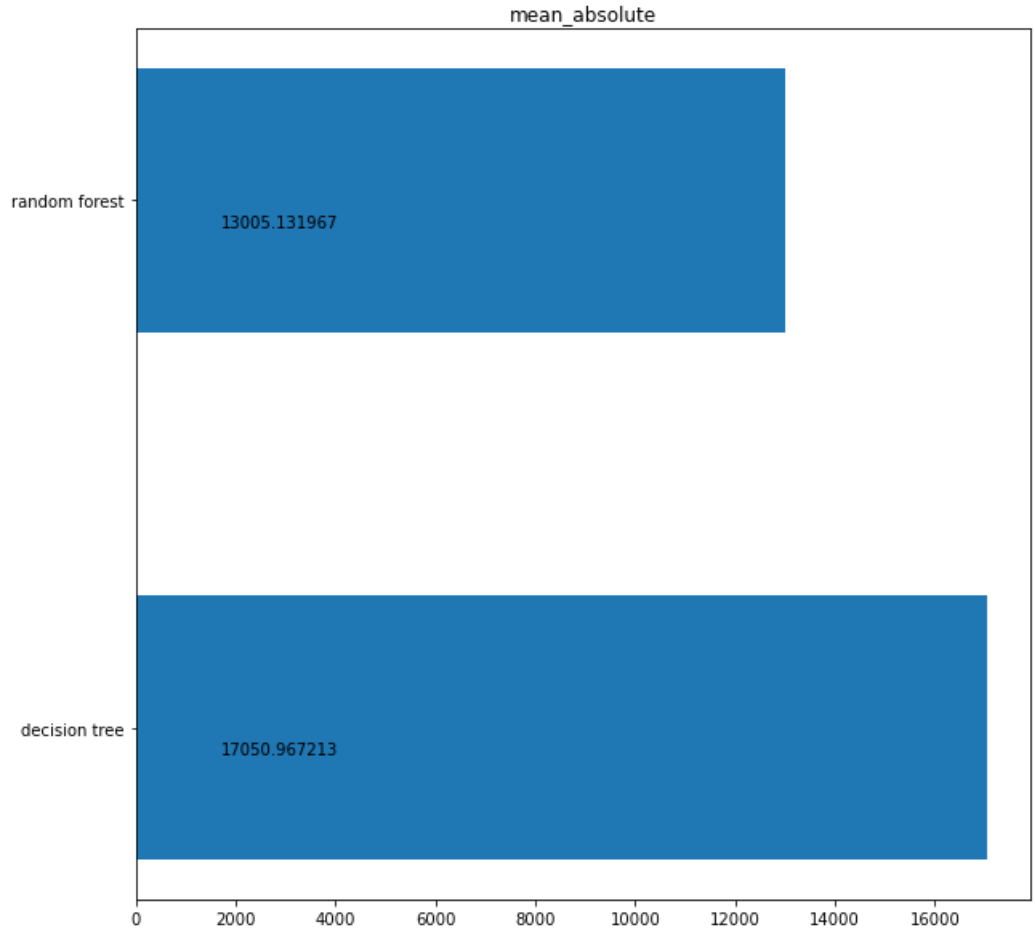
```
def plot_metrics(metrics, models, test_y):
    for name, fun in metrics.items():
        fig, ax = plt.subplots(figsize=(10,10))
        results_metrics = []

        for nm, results in models.items():
            results_metrics.append(fun(test_y, results))

        sorted_el = list(sorted(list(zip(models.keys(), results_metrics)), key=lambda x: -x[1]))
        results_metrics = list(map(lambda x: x[1], sorted_el))
        model_list = list(map(lambda x: x[0], sorted_el))

        pos = np.arange(len(model_list))
        rects = ax.barh(pos, results_metrics,
                        align='center',
                        height=0.5,
                        tick_label=model_list)
        ax.set_title(name)
        for a, b in zip(pos, results_metrics):
            plt.text(max(results_metrics) * 0.1, a-0.05, str(round(b,6)), color='black')
        plt.show()
```

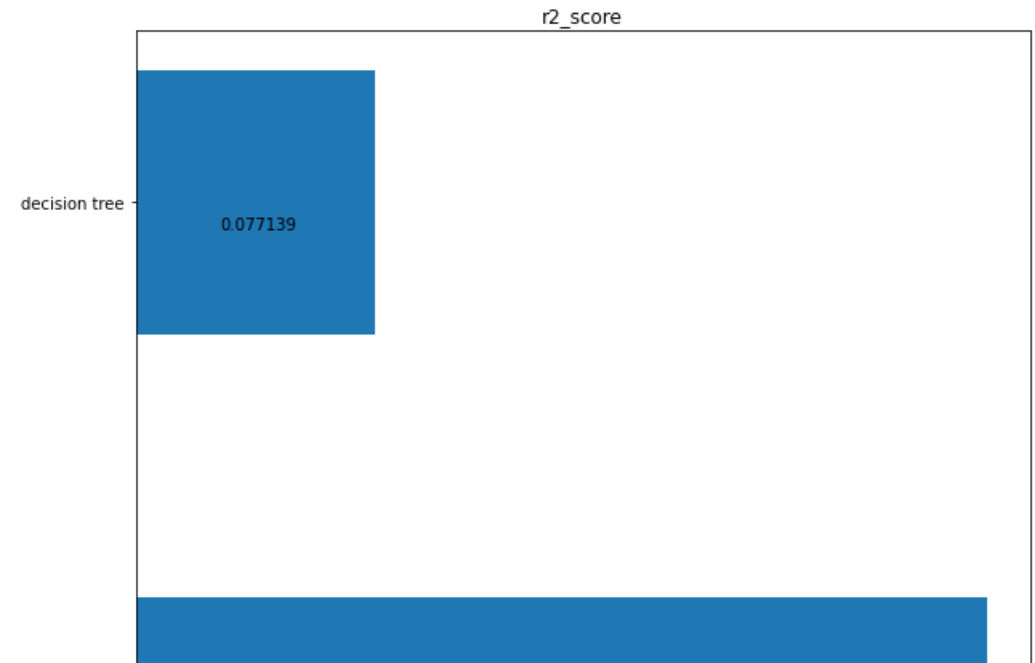
In [15]:

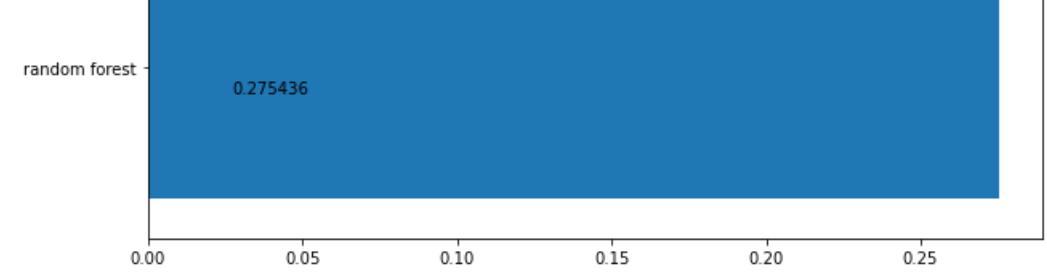
```
metrics = {
    'mean_squared':mean_squared_error,
    'mean_absolute':mean_absolute_error,
    'r2_score':r2_score
}

predictions = {
    'decision tree': tree_predict,
    'random forest': forest_predict
}

plot_metrics(metrics, predictions, df_y_test)
```




decision tree
1193200221.262295
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1e9


mean_absolute
random forest
13005.131967
decision tree
17050.967213
0
2000
4000
6000
8000
10000
12000
14000
16000


r2_score
decision tree
0.077139



Так как данных силшком мало не удалось решить задачу регрессии для признака Salary