Рекомендация тарифов

В вашем распоряжении данные о поведении клиентов, которые уже перешли с архивных тарифов, на новые. Нужно построить модель для задачи классификации, которая выберет подходящий тариф. Предобработка данных не понадобится — вы её уже сделали.

Постройте модель с максимально большим значением *accuracy*. Чтобы сдать проект успешно, нужно довести долю правильных ответов по крайней мере до 0.75. Проверьте *accuracy* на тестовой выборке самостоятельно.

Откройте и изучите файл

In [1]:
```
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from sklearn.metrics import accuracy_score
from statsmodels.stats.outliers_influence import variance_inflation_factor
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
df = pd.read_csv('/home/cookie/projects/users_behavior.csv')
```

In [3]:
```
df.head(10)
```

Out[3]:

	calls	minutes	messages	mb_used	is_ultra
0	40.0	311.90	83.0	19915.42	0
1	85.0	516.75	56.0	22696.96	0
2	77.0	467.66	86.0	21060.45	0
3	106.0	745.53	81.0	8437.39	1
4	66.0	418.74	1.0	14502.75	0
5	58.0	344.56	21.0	15823.37	0
6	57.0	431.64	20.0	3738.90	1
7	15.0	132.40	6.0	21911.60	0
8	7.0	43.39	3.0	2538.67	1
9	90.0	665.41	38.0	17358.61	0

In [4]:
```
df.info()
```

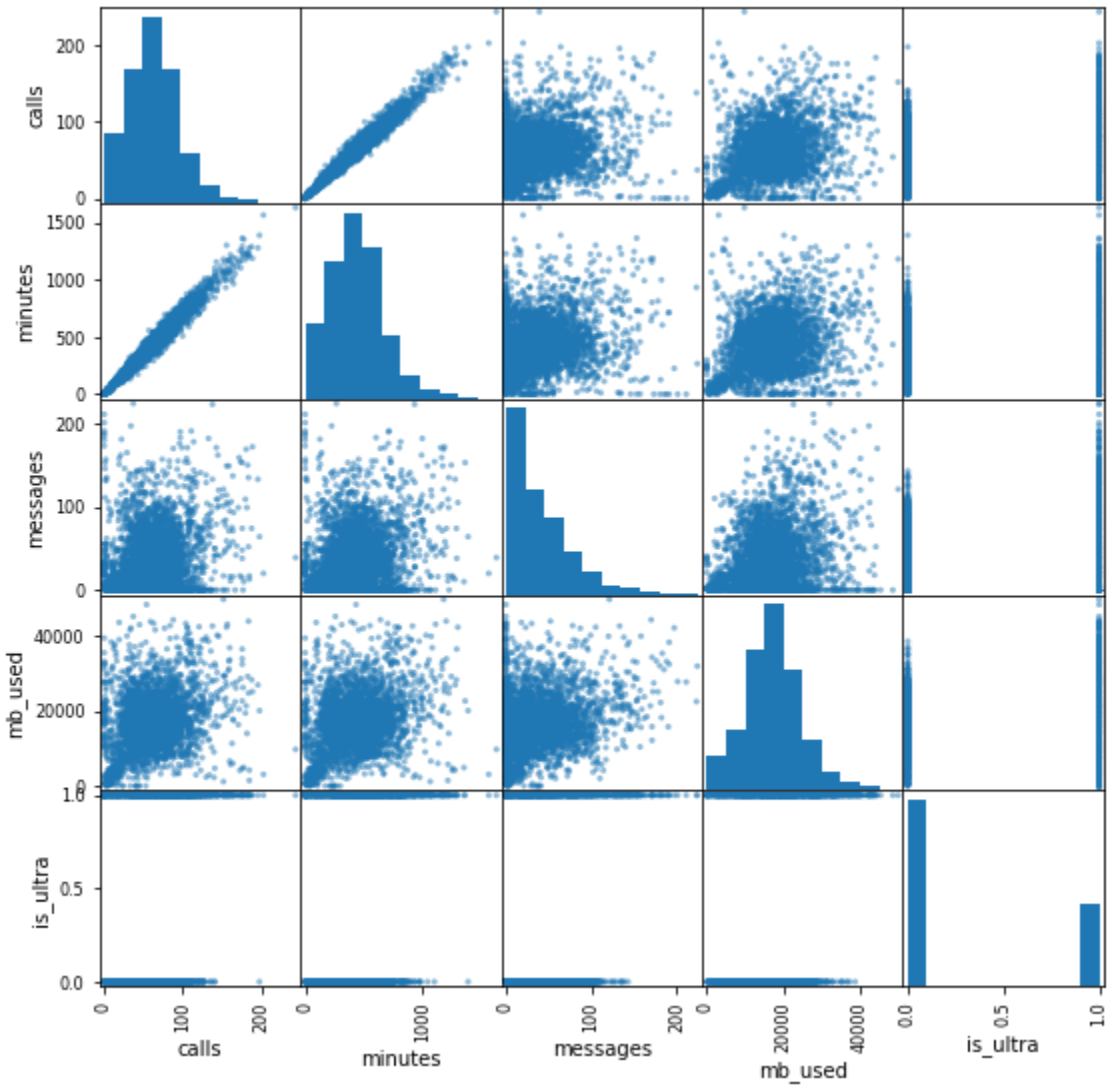
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3214 entries, 0 to 3213
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
```

```
 0   calls     3214 non-null   float64
 1   minutes   3214 non-null   float64
 2   messages  3214 non-null   float64
 3   mb_used   3214 non-null   float64
 4   is_ultra  3214 non-null   int64
dtypes: float64(4), int64(1)
memory usage: 125.7 KB
```

In [5]:
```
test_df = df.copy()
```

In [6]:
```
pd.plotting.scatter_matrix(test_df, figsize=(9, 9))

pass
```



Вывод

У звонков и минут высокая корреляция и надо один из этих столбцов убрать

Разбейте данные на выборки

In [7]:
```
features = df.drop('is_ultra', axis=1)
target = df['is_ultra']
```

In [8]:
```
features_train, features_valid, target_train, target_valid = (
    train_test_split(features, target, test_size=0.20, random_state=12345)
```

```
)
```

In [9]:
```
features_train_2, features_test, target_train_2, target_test = (
    train_test_split(features, target, test_size=.25, random_state=12345)
)
```

Исследуйте модели

Дерево решений

In [10]:
```
for depth in range(1, 6):
    model = DecisionTreeClassifier(random_state=12345, max_depth=depth)
    model.fit(features_train, target_train)
    predictions_valid = model.predict(features_valid)

    print('max_depth =', depth, ': ', end='')
    print(accuracy_score(target_valid, predictions_valid))
```

```
max_depth = 1 : 0.7480559875583204
max_depth = 2 : 0.7807153965785381
max_depth = 3 : 0.7838258164852255
max_depth = 4 : 0.7791601866251944
max_depth = 5 : 0.7853810264385692
```

Вывод

После значения `max_depth = 3` точность модели не стновится лучше

Случайный лес

In [11]:
```
best_model = None
best_result = 0
best_est = 0
for est in range(1, 11):
    model_2 = RandomForestClassifier(random_state=12345, n_estimators=est)
    model_2.fit(features_train, target_train)
    result = model_2.score(features_valid, target_valid)
    if result > best_result:
        best_model = model_2
        best_result = result
        best_est = est
```

In [12]:
```
print("Accuracy наилучшей модели на валидационной выборке:", best_result,"Количество дерев
```

```
Accuracy наилучшей модели на валидационной выборке: 0.7744945567651633 Количество деревье
в: 4
```

Вывод

Количество деревьев 4 дает наибольшую точность

Логистическая регрессия

In [13]:
```
model_3 = LogisticRegression(random_state=12345)
model_3.fit(features_train, target_train)
result = model_3.score(features_valid, target_valid)
```

In [14]:
```
print("Accuracy модели логистической регрессии на валидационной выборке:", result)
```

```
Accuracy модели логистической регрессии на валидационной выборке: 0.7573872472783826
```

Вывод

Логистическая регрессия дает наихудшую точность

Проверьте модель на тестовой выборке

Случайный лес

In [15]:
```
final_model = RandomForestClassifier(random_state=12345, n_estimators=4)
```

In [16]:
```
final_model.fit(features_train, target_train)
```

Out[16]:
```
RandomForestClassifier(n_estimators=4, random_state=12345)
```

In [17]:
```
test_predictions = final_model.predict(features_test)
```

In [18]:
```
def error_count(answers, predictions):
    count = 0
    for i in range(0, len(predictions)):
        if answers[i] != predictions[i]:
            count +=1
    return count
```

In [19]:
```
error_count(target_test.tolist(), test_predictions)
```

Out[19]:
```
156
```

In [20]:
```
def accuracy(answers, predictions):
    correct = 0
    for i in range(0, len(predictions)):
        if answers[i] == predictions[i]:
            correct +=1
    return correct / len(answers)
```

In [21]:
```
accuracy(target_test.tolist(), test_predictions)
```

Out[21]:
```
0.8059701492537313
```

Чек-лист готовности проекта

Поставьте 'x' в выполненных пунктах. Далее нажмите Shift+Enter.

- [x] Jupyter Notebook открыт
- [x] Весь код исполняется без ошибок
- [x] Ячейки с кодом расположены в порядке исполнения
- [x] Выполнено задание 1: данные загружены и изучены

- [x] Выполнено задание 2: данные разбиты на три выборки
- [x] Выполнено задание 3: проведено исследование моделей
  - [x] Рассмотрено больше одной модели
  - [x] Рассмотрено хотя бы 3 значения гипепараметров для какой-нибудь модели
  - [x] Написаны выводы по результатам исследования
- [x] Выполнено задание 3: Проведено тестирование
- [x] Удалось достичь accuracy не меньше 0.75