Московский государственный технический университет им. Н.Э. Баумана
Факультет «Информатика и системы управления»
Кафедра «Системы обработки информации и управления»

Домашнее задание №1
по дисциплине «Методы машинного обучения»
«Решение задачи обучения с учителем»

ИСПОЛНИТЕЛЬ:

Цветкова Алена
Группа ИУ5-21М

_____

ПРЕПОДАВАТЕЛЬ:

Гапанюк Ю.Е.

_____

Москва, 2021

In [7]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator
import seaborn as sns
```

```
//anaconda3/lib/python3.7/site-packages/statsmodels/tools/_testing.py:19: FutureWarning: pandas.util.testing is deprecated. Use the functions in the public API at pandas.testing instead.
  import pandas.util.testing as tm
```

Данные

In [8]:

```
data = pd.read_csv('auto-mpg.csv')
data_raw = pd.read_csv('auto-mpg.csv')
data.head()
```

Out[8]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin
0	18.0	8	307.0	130	3504	12.0	70	1
1	15.0	8	350.0	165	3693	11.5	70	1
2	18.0	8	318.0	150	3436	11.0	70	1
3	16.0	8	304.0	150	3433	12.0	70	1
4	17.0	8	302.0	140	3449	10.5	70	1

Замена ? на пропуски и пропусков на медиану

In [9]:

```
data["horsepower"].replace({"?": np.nan}, inplace=True)
```

In [10]:

```
data.isnull().sum()
```

Out[10]:

```
mpg             0
```

```
cylinders       0
displacement    0
horsepower      6

weight          0
acceleration    0
model year      0
origin          0
car name        0
dtype: int64
```

In [11]:

```
def impute_column(dataset, column, strategy_param, fill_value_param=None):
    """
    Заполнение пропусков в одном признаке
    """
    temp_data = dataset[[column]].values
    size = temp_data.shape[0]

    indicator = MissingIndicator()
    mask_missing_values_only = indicator.fit_transform(temp_data)

    imputer = SimpleImputer(strategy=strategy_param,
                            fill_value=fill_value_param)
    all_data = imputer.fit_transform(temp_data)

    missed_data = temp_data[mask_missing_values_only]
    filled_data = all_data[mask_missing_values_only]

    return all_data.reshape((size,)), filled_data, missed_data
```

In [12]:

```
all_data, filled_data, missed_data = impute_column(data, 'horsepower', 'me
dian')
```

In [13]:

```
data['horsepower']= all_data
```

Теперь нет пропусков

In [14]:

```
data.isnull().sum()
```

Out[14]:

```
mpg             0
cylinders       0
displacement    0
horsepower      0
weight          0
acceleration    0
model year      0
origin          0
car name        0
dtype: int64
```

В наборе нет категориальных признаков:

In [15]:

```
data.dtypes
```

Out[15]:

```
mpg              float64
cylinders          int64
displacement     float64
horsepower       float64
weight             int64
acceleration     float64
model year         int64
origin             int64
car name          object
dtype: object
```

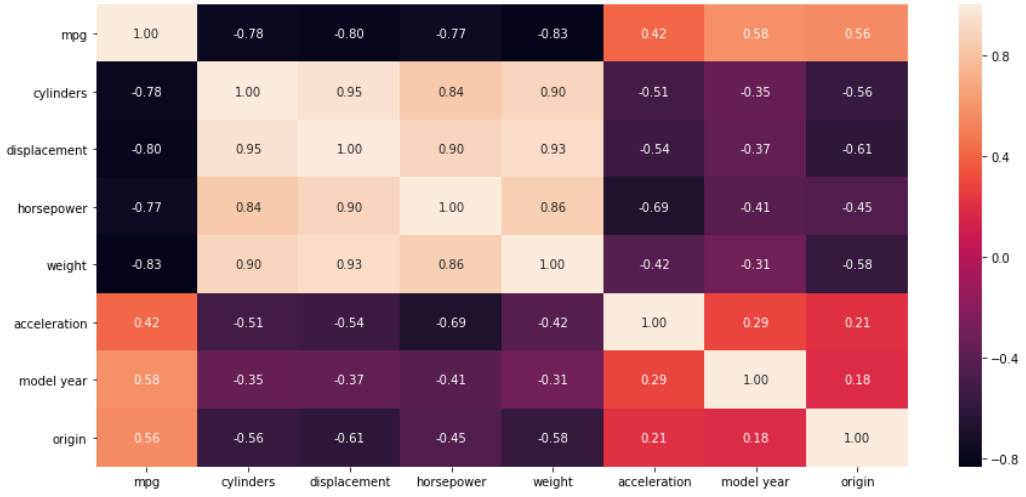
mpg - целевой признак

In [16]:

```
fig, ax = plt.subplots(figsize=(15,7))
sns.heatmap(data.corr(method='pearson'), ax=ax, annot=True, fmt='.2f')
```

Out[16]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x130045978>
```



Инвертируем для получения положительной корелляции

In [17]:

```
data['inv cylinders'] = 0 - data['cylinders']
data['inv displacement'] = 0 - data['displacement']
data['inv weight'] = 0 - data['weight']
data['inv horsepower'] = 0 - data['horsepower']
```

In [18]:

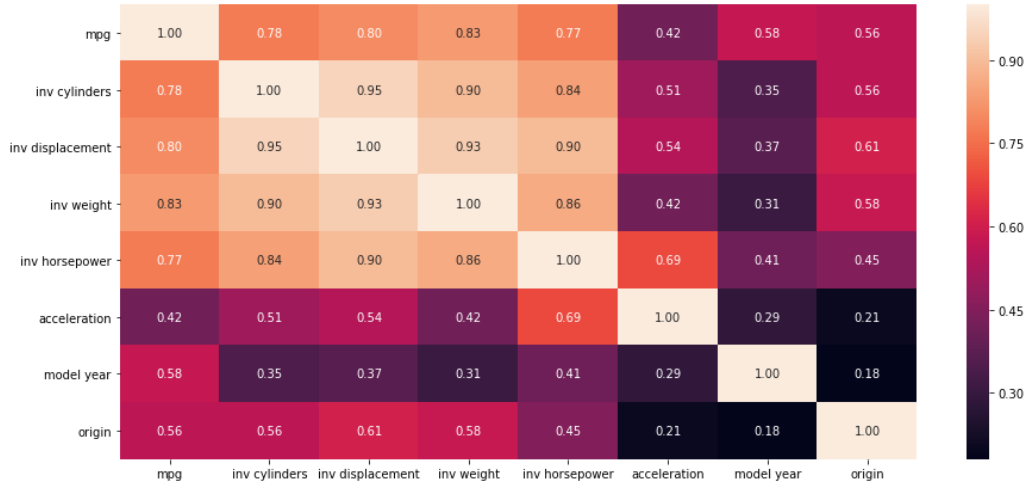
```
data = data [['mpg', 'inv cylinders', 'inv displacement', 'inv weight', 'i
nv horsepower', 'acceleration', 'model year', 'origin']]
```

In [19]:

```
fig, ax = plt.subplots(figsize=(15,7))
sns.heatmap(data.corr(method='pearson'), ax=ax, annot=True, fmt='.2f')
```

Out[19]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x130278b38>
```



Масштабирование

In [20]:

```
from sklearn.preprocessing import MinMaxScaler
mmc = MinMaxScaler()
data[['inv cylinders']] = mmc.fit_transform(data[['inv cylinders']])
data[['inv displacement']] = mmc.fit_transform(data[['inv displacement']])
data[['inv weight']] = mmc.fit_transform(data[['inv weight']])
data[['inv horsepower']] = mmc.fit_transform(data[['inv horsepower']])
data[['acceleration']] = mmc.fit_transform(data[['acceleration']])
data[['model year']] = mmc.fit_transform(data[['model year']])
data[['origin']] = mmc.fit_transform(data[['origin']])
data[['mpg']] = mmc.fit_transform(data[['mpg']])
```

In [21]:

```
data.describe()
```

Out[21]:

	mpg	inv cylinders	inv displacement	inv weight	inv horsepower	acceleration
count	398.000000	398.000000	398.000000	398.000000	398.000000	398.000000
mean	0.386026	0.509045	0.675902	0.615133	0.683130	0.450482
std	0.207872	0.340201	0.269431	0.240103	0.207732	0.164148
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.226064	0.000000	0.498708	0.434363	0.570652	0.346726

50%	0.372340	0.800000	0.791990	0.662461	0.741848	0.446429
75%	0.531915	0.800000	0.906331	0.826836	0.836957	0.546131
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

Отбор признаков, наиболее подходящих для построения модели

In [22]:

```
from sklearn.neighbors import KNeighborsRegressor
from mlxtend.feature_selection import ExhaustiveFeatureSelector as EFS
```

In [23]:

```
data_x = data[['inv cylinders', 'inv displacement', 'inv weight', 'inv hor
sepower', 'acceleration', 'model year', 'origin']]
data_y = data['mpg']
```

In [24]:

```
knn = KNeighborsRegressor(n_neighbors=3)
```

In [29]:

```
%%time
efs1 = EFS(knn,
           min_features=2,
           max_features=7,
           scoring='neg_mean_squared_error',
           print_progress=True,
           cv=5)

efs1 = efs1.fit(data_x, data_y, custom_feature_names=data_x.columns)

#print('Best accuracy score: %.2f' % efs1.best_score_)
print('Best subset (indices):', efs1.best_idx_)
print('Best subset (corresponding names):', efs1.best_feature_names_)
```

```
Features: 120/120
```

```
Best subset (indices): (1, 2, 3, 5, 6)
Best subset (corresponding names): ('inv displacement', 'inv
weight', 'inv horsepower', 'model year', 'origin')
CPU times: user 1.47 s, sys: 49.4 ms, total: 1.52 s
Wall time: 1.59 s
```

Разделение выборки

In [25]:

```
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import LeaveOneOut
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error, mean_
squared_log_error, median_absolute_error, r2_score
```

In [26]:

```
data_X_train, data_X_test, data_y_train, data_y_test = train_test_split(
    data_x, data_y, test_size=0.3, random_state=1)
data_X_train.shape, data_X_test.shape
```

Out[26]:

```
((278, 7), (120, 7))
```

Подбор гиперпараметров и KNN

In [27]:

```
n_range = np.array(range(2,55,1))
tuned_parameters = [{'n_neighbors': n_range}]
tuned_parameters
```

Out[27]:

```
[{'n_neighbors': array([ 2,  3,  4,  5,  6,  7,  8,  9, 10, 1
1, 12, 13, 14, 15, 16, 17, 18,
         19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31,
32, 33, 34, 35,
         36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48,
49, 50, 51, 52,
         53, 54])}]
```

In [35]:

```
%%time
clf_gs = GridSearchCV(KNeighborsRegressor(), tuned_parameters, cv=LeaveOne
Out(), scoring='neg_mean_squared_error')
clf_gs.fit(data_X_train, data_y_train)
```

```
CPU times: user 1min 12s, sys: 711 ms, total: 1min 12s
Wall time: 1min 23s
```

Out[35]:

```
GridSearchCV(cv=LeaveOneOut(), error_score='raise-deprecatin
g',
             estimator=KNeighborsRegressor(algorithm='auto',
leaf_size=30,
                                           metric='minkowsk
i',
                                           metric_params=Non
e, n_jobs=None,
                                           n_neighbors=5, p=
2,
                                           weights='unifor
m'),
             iid='warn', n_jobs=None,
             param_grid=[{'n_neighbors': array([ 2,  3,  4,
5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
       19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 3
2, 33, 34, 35,
       36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 4
9, 50, 51, 52,
```

```
       53, 54])}],
             pre_dispatch='2*n_jobs', refit=True, return_trai
n_score=False,
             scoring='neg_mean_squared_error', verbose=0)
```

In [36]:

```
clf_gs.best_params_
```

Out[36]:

```
{'n_neighbors': 3}
```

In [28]:

```
KNN = KNeighborsRegressor(n_neighbors=3)
KNN.fit(data_X_train, data_y_train)
target_KNN = KNN.predict(data_X_test)
```

In [29]:

```
mean_absolute_error(data_y_test, target_KNN)
```

Out[29]:

```
0.057143912529550825
```

In [30]:

```
median_absolute_error(data_y_test, target_KNN)
```

Out[30]:

```
0.03989361702127657
```

Autogluon

In [31]:

```
from autogluon.tabular import TabularPredictor
```

In [33]:

```
data_train, data_test = train_test_split(
    data, test_size=0.3, random_state=1)
data_train.shape, data_test.shape
```

Out[33]:

```
((278, 8), (120, 8))
```

In [34]:

```
predictor = TabularPredictor(label='mpg').fit(train_data=data_train)
```

```
No path specified. Models will be saved in: "AutogluonModels/
ag-20210614_182648/"
Beginning AutoGluon training ...
AutoGluon will save models to "AutogluonModels/ag-20210614_18
2648/"
AutoGluon Version:  0.2.0
```

```
AutoGluon Version:  0.2.0
Train Data Rows:    278
Train Data Columns: 7
Preprocessing data ...
AutoGluon infers your prediction problem is: 'regression' (be
cause dtype of label-column == float and many unique label-va
lues observed).
        Label info (max, min, mean, stddev): (1.0000000000000
002, 0.0, 0.38832, 0.20975)
        If 'regression' is not the correct problem_type, plea
se manually specify the problem_type argument in fit() (You m
ay specify problem_type as one of: ['binary', 'multiclass',
'regression'])
Using Feature Generators to preprocess the data ...
Fitting AutoMLPipelineFeatureGenerator...
        Available Memory:                    2336.57 MB
        Train Data (Original)  Memory Usage: 0.02 MB (0.0% of
available memory)
        Inferring data type of each feature based on column v
alues. Set feature_metadata_in to manually specify special dt
ypes of the features.
        Stage 1 Generators:
                Fitting AsTypeFeatureGenerator...
        Stage 2 Generators:
                Fitting FillNaFeatureGenerator...
        Stage 3 Generators:
                Fitting IdentityFeatureGenerator...
        Stage 4 Generators:
                Fitting DropUniqueFeatureGenerator...
        Types of features in original data (raw dtype, specia
l dtypes):
                ('float', []) : 7 | ['inv cylinders', 'inv di
splacement', 'inv weight', 'inv horsepower', 'acceleration',
...]
        Types of features in processed data (raw dtype, speci
al dtypes):
                ('float', []) : 7 | ['inv cylinders', 'inv di
splacement', 'inv weight', 'inv horsepower', 'acceleration',
...]
        0.1s = Fit runtime
        7 features in original data used to generate 7 featur
es in processed data.
        Train Data (Processed) Memory Usage: 0.02 MB (0.0% of
available memory)
Data preprocessing and feature engineering runtime = 0.15s
...
AutoGluon will gauge predictive performance using evaluation
metric: 'root_mean_squared_error'
        To change this, specify the eval_metric argument of f
it()
Automatically generating train/validation split with holdout_
frac=0.2, Train Rows: 222, Val Rows: 56
Fitting model: KNeighborsUnif ...
        -0.0918  = Validation root_mean_squared_error score
        0.01s    = Training runtime
        0.12s    = Validation runtime
Fitting model: KNeighborsDist ...
        -0.088   = Validation root_mean_squared_error score
        0.01s    = Training runtime
        0.11s    = Validation runtime
Fitting model: LightGBMXT
```

```
Fitting model: LightGBMXT ...
        Warning: Exception caused LightGBMXT to fail during t
raining (ImportError)... Skipping this model.
                `import lightgbm` failed. If you are using Ma
c OSX, Please try 'brew install libomp'. Detailed info: dlope
n(//anaconda3/lib/python3.7/site-packages/lightgbm/lib_lightg
bm.so, 6): Library not loaded: /usr/local/opt/libomp/lib/libo
mp.dylib
  Referenced from: //anaconda3/lib/python3.7/site-packages/li
ghtgbm/lib_lightgbm.so
  Reason: image not found
Fitting model: LightGBM ...
        Warning: Exception caused LightGBM to fail during tra
ining (ImportError)... Skipping this model.
                `import lightgbm` failed. If you are using Ma
c OSX, Please try 'brew install libomp'. Detailed info: dlope
n(//anaconda3/lib/python3.7/site-packages/lightgbm/lib_lightg
bm.so, 6): Library not loaded: /usr/local/opt/libomp/lib/libo
mp.dylib
  Referenced from: //anaconda3/lib/python3.7/site-packages/li
ghtgbm/lib_lightgbm.so
  Reason: image not found
Fitting model: RandomForestMSE ...
        -0.0793  = Validation root_mean_squared_error score
        0.75s    = Training runtime
        0.11s    = Validation runtime
Fitting model: CatBoost ...
        -0.083   = Validation root_mean_squared_error score
        0.67s    = Training runtime
        0.01s    = Validation runtime
Fitting model: ExtraTreesMSE ...
        -0.0847  = Validation root_mean_squared_error score
        0.72s    = Training runtime
        0.11s    = Validation runtime
Fitting model: NeuralNetFastAI ...
        -0.0781  = Validation root_mean_squared_error score
        4.7s     = Training runtime
        0.03s    = Validation runtime
Fitting model: XGBoost ...
        Warning: Exception caused XGBoost to fail during trai
ning... Skipping this model.
                XGBoost Library (libxgboost.dylib) could not
be loaded.
Likely causes:
  * OpenMP runtime is not installed (vcomp140.dll or libgomp-
1.dll for Windows, libomp.dylib for Mac OSX, libgomp.so for L
inux and other UNIX-like OSes). Mac OSX users: Run `brew inst
all libomp` to install OpenMP runtime.
  * You are running 32-bit Python on a 64-bit OS
Error message(s): ['dlopen(//anaconda3/lib/python3.7/site-pac
kages/xgboost/lib/libxgboost.dylib, 6): Library not loaded: /
usr/local/opt/libomp/lib/libomp.dylib\n  Referenced from: //a
naconda3/lib/python3.7/site-packages/xgboost/lib/libxgboost.d
ylib\n  Reason: image not found']

Detailed Traceback:
Traceback (most recent call last):
  File "//anaconda3/lib/python3.7/site-packages/autogluon/tab
ular/trainer/abstract_trainer.py", line 924, in _train_and_sa
ve
    model = self._train_single(X, y, model, X_val, y_val, **m
```

```
    model = self._train_single(X, y, model, X_val, y_val, **m
odel_fit_kwargs)
  File "//anaconda3/lib/python3.7/site-packages/autogluon/tab
ular/trainer/abstract_trainer.py", line 896, in _train_single
    model.fit(X=X, y=y, X_val=X_val, y_val=y_val, **model_fit
_kwargs)
  File "//anaconda3/lib/python3.7/site-packages/autogluon/cor
e/models/abstract/abstract_model.py", line 411, in fit
    self._fit(**kwargs)
  File "//anaconda3/lib/python3.7/site-packages/autogluon/tab
ular/models/xgboost/xgboost_model.py", line 116, in _fit
    try_import_xgboost()
  File "//anaconda3/lib/python3.7/site-packages/autogluon/cor
e/utils/try_import.py", line 82, in try_import_xgboost
    import xgboost
  File "//anaconda3/lib/python3.7/site-packages/xgboost/__ini
t__.py", line 9, in <module>
    from .core import DMatrix, DeviceQuantileDMatrix, Booster
  File "//anaconda3/lib/python3.7/site-packages/xgboost/core.
py", line 174, in <module>
    _LIB = _load_lib()
  File "//anaconda3/lib/python3.7/site-packages/xgboost/core.
py", line 165, in _load_lib
    'Error message(s): {}\n'.format(os_error_list))
xgboost.core.XGBoostError: XGBoost Library (libxgboost.dylib)
could not be loaded.
Likely causes:
  * OpenMP runtime is not installed (vcomp140.dll or libgomp-
1.dll for Windows, libomp.dylib for Mac OSX, libgomp.so for L
inux and other UNIX-like OSes). Mac OSX users: Run `brew inst
all libomp` to install OpenMP runtime.
  * You are running 32-bit Python on a 64-bit OS
Error message(s): ['dlopen(//anaconda3/lib/python3.7/site-pac
kages/xgboost/lib/libxgboost.dylib, 6): Library not loaded: /
usr/local/opt/libomp/lib/libomp.dylib\n  Referenced from: //a
naconda3/lib/python3.7/site-packages/xgboost/lib/libxgboost.d
ylib\n  Reason: image not found']

Fitting model: NeuralNetMXNet ...
        Warning: Exception caused NeuralNetMXNet to fail duri
ng training (ImportError)... Skipping this model.
                Unable to import dependency mxnet. A quick ti
p is to install via `pip install mxnet --upgrade`, or `pip in
stall mxnet_cu101 --upgrade`
Fitting model: LightGBMLarge ...
        Warning: Exception caused LightGBMLarge to fail durin
g training (ImportError)... Skipping this model.
                `import lightgbm` failed. If you are using Ma
c OSX, Please try 'brew install libomp'. Detailed info: dlope
n(//anaconda3/lib/python3.7/site-packages/lightgbm/lib_lightg
bm.so, 6): Library not loaded: /usr/local/opt/libomp/lib/libo
mp.dylib
  Referenced from: //anaconda3/lib/python3.7/site-packages/li
ghtgbm/lib_lightgbm.so
  Reason: image not found
Fitting model: WeightedEnsemble_L2 ...
        -0.0735  = Validation root_mean_squared_error score
        0.33s    = Training runtime
        0.0s     = Validation runtime
AutoGluon training complete, total runtime = 9.49s ...
TabularPredictor saved. To load, use: predictor = TabularPred
```

```
ictor.load("AutogluonModels/ag-20210614_182648/")
```

In [35]:

```
predictions = predictor.predict(data_test)
```

In [45]:

```
perf = predictor.evaluate_predictions(y_true=data_test['mpg'], y_pred=pred
ictions, auxiliary_metrics=True)
```

```
Evaluation: root_mean_squared_error on test data: -0.07356098
44055441
        Note: Scores are always higher_is_better. This metric
score can be multiplied by -1 to get the metric value.
Evaluations on test data:
{
    "root_mean_squared_error": -0.0735609844055441,
    "mean_squared_error": -0.005411218426712701,
    "mean_absolute_error": -0.055652427647579855,
    "r2": 0.8691540220016181,
    "pearsonr": 0.938895424672481,
    "median_absolute_error": -0.04143354835662434
}
```

KNN на сырых данных

In [62]:

```
data_raw["horsepower"].replace({"?": np.nan}, inplace=True)
data_raw[['mpg']] = mmc.fit_transform(data_raw[['mpg']])
```

In [63]:

```
data_raw = data_raw.dropna(axis=0, how='any')
data_raw.shape
```

Out[63]:

```
(392, 9)
```

In [64]:

```
data_raw_x = data_raw[['cylinders', 'displacement', 'weight', 'horsepower'
, 'acceleration', 'model year', 'origin']]
data_raw_y = data_raw['mpg']
```

In [65]:

```
data_raw_X_train, data_raw_X_test, data_raw_y_train, data_raw_y_test = tra
in_test_split(
    data_raw_x, data_raw_y, test_size=0.3, random_state=1)
data_raw_X_train.shape, data_raw_X_test.shape
```

Out[65]:

```
((274, 7), (118, 7))
```

In [66]:

```
KNN = KNeighborsRegressor(n_neighbors=3)
KNN.fit(data_raw_X_train, data_raw_y_train)
target_KNN = KNN.predict(data_raw_X_test)
```

In [67]:

```
mean_absolute_error(data_raw_y_test, target_KNN)
```

Out[67]:

```
0.0850387666786873
```

In [69]:

```
median_absolute_error(data_raw_y_test, target_KNN)
```

Out[69]:

```
0.0642730496453901
```

Autogluon на сырых данных

In [70]:

```
data_raw_train, data_raw_test = train_test_split(
    data_raw, test_size=0.3, random_state=1)
```

In [71]:

```
predictor = TabularPredictor(label='mpg').fit(train_data=data_raw_train)
```

```
No path specified. Models will be saved in: "AutogluonModels/
ag-20210614_190153/"
Beginning AutoGluon training ...
AutoGluon will save models to "AutogluonModels/ag-20210614_19
0153/"
AutoGluon Version:  0.2.0
Train Data Rows:    274
Train Data Columns: 8
Preprocessing data ...
AutoGluon infers your prediction problem is: 'regression' (be
cause dtype of label-column == float and many unique label-va
lues observed).
        Label info (max, min, mean, stddev): (1.0000000000000
002, 0.0, 0.38335, 0.20343)
        If 'regression' is not the correct problem_type, plea
se manually specify the problem_type argument in fit() (You m
ay specify problem_type as one of: ['binary', 'multiclass',
'regression'])
Using Feature Generators to preprocess the data ...
Fitting AutoMLPipelineFeatureGenerator...
        Available Memory:                    2481.69 MB
        Train Data (Original)  Memory Usage: 0.05 MB (0.0% of
available memory)
        Inferring data type of each feature based on column v
alues. Set feature_metadata_in to manually specify special dt
ypes of the features.
```

```
        Stage 1 Generators:
                Fitting AsTypeFeatureGenerator...
        Stage 2 Generators:

                Fitting FillNaFeatureGenerator...
        Stage 3 Generators:
                Fitting IdentityFeatureGenerator...
                Fitting CategoryFeatureGenerator...
                        Fitting CategoryMemoryMinimizeFeature
Generator...
        Stage 4 Generators:
                Fitting DropUniqueFeatureGenerator...
        Types of features in original data (raw dtype, specia
l dtypes):
                ('float', [])  : 2 | ['displacement', 'accele
ration']
                ('int', [])    : 4 | ['cylinders', 'weight',
'model year', 'origin']
                ('object', []) : 2 | ['horsepower', 'car nam
e']
        Types of features in processed data (raw dtype, speci
al dtypes):
                ('category', []) : 2 | ['horsepower', 'car na
me']
                ('float', [])    : 2 | ['displacement', 'acce
leration']
                ('int', [])      : 4 | ['cylinders', 'weigh
t', 'model year', 'origin']
        0.1s = Fit runtime
        8 features in original data used to generate 8 featur
es in processed data.
        Train Data (Processed) Memory Usage: 0.02 MB (0.0% of
available memory)
Data preprocessing and feature engineering runtime = 0.19s
...
AutoGluon will gauge predictive performance using evaluation
metric: 'root_mean_squared_error'
        To change this, specify the eval_metric argument of f
it()
Automatically generating train/validation split with holdout_
frac=0.2, Train Rows: 219, Val Rows: 55
Fitting model: KNeighborsUnif ...
        -0.1124  = Validation root_mean_squared_error score
        0.0s     = Training runtime
        0.12s    = Validation runtime
Fitting model: KNeighborsDist ...
        -0.1056  = Validation root_mean_squared_error score
        0.01s    = Training runtime
        0.11s    = Validation runtime
Fitting model: LightGBMXT ...
        Warning: Exception caused LightGBMXT to fail during t
raining (ImportError)... Skipping this model.
                `import lightgbm` failed. If you are using Ma
c OSX, Please try 'brew install libomp'. Detailed info: dlope
n(//anaconda3/lib/python3.7/site-packages/lightgbm/lib_lightg
bm.so, 6): Library not loaded: /usr/local/opt/libomp/lib/libo
mp.dylib
  Referenced from: //anaconda3/lib/python3.7/site-packages/li
ghtgbm/lib_lightgbm.so
  Reason: image not found
Fitting model: LightGBM
```

```
Fitting model: LightGBM ...
        Warning: Exception caused LightGBM to fail during tra
ining (ImportError)... Skipping this model.
                `import lightgbm` failed. If you are using Ma
c OSX, Please try 'brew install libomp'. Detailed info: dlope
n(//anaconda3/lib/python3.7/site-packages/lightgbm/lib_lightg
bm.so, 6): Library not loaded: /usr/local/opt/libomp/lib/libo
mp.dylib
  Referenced from: //anaconda3/lib/python3.7/site-packages/li
ghtgbm/lib_lightgbm.so
  Reason: image not found
Fitting model: RandomForestMSE ...
        -0.0769  = Validation root_mean_squared_error score
        0.94s    = Training runtime
        0.11s    = Validation runtime
Fitting model: CatBoost ...
        -0.0664  = Validation root_mean_squared_error score
        0.91s    = Training runtime
        0.01s    = Validation runtime
Fitting model: ExtraTreesMSE ...
        -0.0692  = Validation root_mean_squared_error score
        0.78s    = Training runtime
        0.11s    = Validation runtime
Fitting model: NeuralNetFastAI ...
        -0.0605  = Validation root_mean_squared_error score
        3.8s     = Training runtime
        0.04s    = Validation runtime
Fitting model: XGBoost ...
        Warning: Exception caused XGBoost to fail during trai
ning... Skipping this model.
                XGBoost Library (libxgboost.dylib) could not
be loaded.
Likely causes:
  * OpenMP runtime is not installed (vcomp140.dll or libgomp-
1.dll for Windows, libomp.dylib for Mac OSX, libgomp.so for L
inux and other UNIX-like OSes). Mac OSX users: Run `brew inst
all libomp` to install OpenMP runtime.
  * You are running 32-bit Python on a 64-bit OS
Error message(s): ['dlopen(//anaconda3/lib/python3.7/site-pac
kages/xgboost/lib/libxgboost.dylib, 6): Library not loaded: /
usr/local/opt/libomp/lib/libomp.dylib\n  Referenced from: //a
naconda3/lib/python3.7/site-packages/xgboost/lib/libxgboost.d
ylib\n  Reason: image not found']

Detailed Traceback:
Traceback (most recent call last):
  File "//anaconda3/lib/python3.7/site-packages/autogluon/tab
ular/trainer/abstract_trainer.py", line 924, in _train_and_sa
ve
    model = self._train_single(X, y, model, X_val, y_val, **m
odel_fit_kwargs)
  File "//anaconda3/lib/python3.7/site-packages/autogluon/tab
ular/trainer/abstract_trainer.py", line 896, in _train_single
    model.fit(X=X, y=y, X_val=X_val, y_val=y_val, **model_fit
_kwargs)
  File "//anaconda3/lib/python3.7/site-packages/autogluon/cor
e/models/abstract/abstract_model.py", line 411, in fit
    self._fit(**kwargs)
  File "//anaconda3/lib/python3.7/site-packages/autogluon/tab
ular/models/xgboost/xgboost_model.py", line 116, in _fit
    try_import_xgboost()
```

```
    try_import_xgboost()
  File "//anaconda3/lib/python3.7/site-packages/autogluon/core/utils/try_import.py", line 82, in try_import_xgboost
    import xgboost
  File "//anaconda3/lib/python3.7/site-packages/xgboost/__init__.py", line 9, in <module>
    from .core import DMatrix, DeviceQuantileDMatrix, Booster
  File "//anaconda3/lib/python3.7/site-packages/xgboost/core.py", line 174, in <module>
    _LIB = _load_lib()
  File "//anaconda3/lib/python3.7/site-packages/xgboost/core.py", line 165, in _load_lib
    'Error message(s): {}\n'.format(os_error_list))
xgboost.core.XGBoostError: XGBoost Library (libxgboost.dylib) could not be loaded.
Likely causes:
  * OpenMP runtime is not installed (vcomp140.dll or libgomp-1.dll for Windows, libomp.dylib for Mac OSX, libgomp.so for Linux and other UNIX-like OSes). Mac OSX users: Run `brew install libomp` to install OpenMP runtime.
  * You are running 32-bit Python on a 64-bit OS
Error message(s): ['dlopen(//anaconda3/lib/python3.7/site-packages/xgboost/lib/libxgboost.dylib, 6): Library not loaded: /usr/local/opt/libomp/lib/libomp.dylib\n  Referenced from: //anaconda3/lib/python3.7/site-packages/xgboost/lib/libxgboost.dylib\n  Reason: image not found']

Fitting model: NeuralNetMXNet ...
        Warning: Exception caused NeuralNetMXNet to fail during training (ImportError)... Skipping this model.
                Unable to import dependency mxnet. A quick tip is to install via `pip install mxnet --upgrade`, or `pip install mxnet_cu101 --upgrade`
Fitting model: LightGBMLarge ...
        Warning: Exception caused LightGBMLarge to fail during training (ImportError)... Skipping this model.
                `import lightgbm` failed. If you are using Mac OSX, Please try 'brew install libomp'. Detailed info: dlopen(//anaconda3/lib/python3.7/site-packages/lightgbm/lib_lightgbm.so, 6): Library not loaded: /usr/local/opt/libomp/lib/libomp.dylib
  Referenced from: //anaconda3/lib/python3.7/site-packages/lightgbm/lib_lightgbm.so
  Reason: image not found
Fitting model: WeightedEnsemble_L2 ...
        -0.0569  = Validation root_mean_squared_error score
        0.4s     = Training runtime
        0.0s     = Validation runtime
AutoGluon training complete, total runtime = 8.41s ...
TabularPredictor saved. To load, use: predictor = TabularPredictor.load("AutogluonModels/ag-20210614_190153/")
```

In [72]:

```
predictions = predictor.predict(data_raw_test)
```

In [73]:

```
perf = predictor.evaluate_predictions(y_true=data_raw_test['mpg'], y_pred=predictions, auxiliary_metrics=True)
```

```
Evaluation: root_mean_squared_error on test data: -0.08133866
038767745
        Note: Scores are always higher_is_better. This metric
score can be multiplied by -1 to get the metric value.
Evaluations on test data:
{
    "root_mean_squared_error": -0.08133866038767745,
    "mean_squared_error": -0.0066159776736619295,
    "mean_absolute_error": -0.058148219517818046,
    "r2": 0.859312359334793,
    "pearsonr": 0.9303026577827065,
    "median_absolute_error": -0.0427735460565446
}
```

Вывод

После предобработки данных средняя и медианная ошибка составили **0.057143912529550825** и **0.03989361702127657** для KNN и **0.055652427647579855** и **0.04143354835662434** для Autogluon. Без предобработки данных средняя и медианная ошибка увеличивается: **0.0850387666786873** и **0.0642730496453901** для KNN и **0.08133866038767745** и **0.0427735460565446** для Autogluon

In [ ]: