ORCA 4500 - Foundations of Data Science (Homework 5)

In [1]:

```
import pandas as pd
import numpy as np
from sklearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV, train_test_split
from sklearn.preprocessing import StandardScaler, PolynomialFeatures,OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.linear_model import Ridge
from sklearn.ensemble import RandomForestRegressor
```

In [2]:

```
df = pd.read_csv('bb_agg.csv').drop(['Unnamed: 0'],axis=1)
```

In [3]:

```
df
```

Out[3]:

	yearID	teamID	H_bat	2B_bat	3B_bat	HR_bat	BB_bat	SB_bat	CS_bat	GIDP_bat	HR_ptch	BB_ptch	SO_ptch	H_ptch
0	1997	ANA	1531	279	25	161	617	126.0	72.0	129.0	202	605	1050	150
1	1997	ATL	1490	268	37	174	597	108.0	58.0	143.0	111	450	1196	131
2	1997	BAL	1498	264	22	196	586	63.0	26.0	121.0	164	563	1139	140
3	1997	BOS	1684	373	32	185	514	68.0	48.0	155.0	149	611	987	156
4	1997	CHA	1498	260	28	158	569	106.0	52.0	133.0	175	575	961	150
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
653	2018	SLN	1369	248	9	205	525	63.0	32.0	92.0	144	593	1337	135
654	2018	TBA	1415	274	43	150	540	128.0	51.0	122.0	164	501	1421	123
655	2018	TEX	1308	266	24	194	555	74.0	35.0	104.0	222	491	1121	151
656	2018	TOR	1336	320	16	217	499	47.0	30.0	118.0	208	551	1298	147
657	2018	WAS	1402	284	25	191	631	119.0	33.0	104.0	198	487	1417	132

658 rows × 17 columns

In [4]:

```
y = df['W']
X = df.drop(['W','3B_bat','BB_bat','G','teamID'],axis=1)
```

In [5]:

```
cat_var = ['yearID','lgID']
cont_var = X.columns.difference(cat_var)

# Is this the right place to split the data ?
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=42)
```

In [6]:

```
pipe_cat_1 = Pipeline([('onehot',OneHotEncoder(handle_unknown = 'ignore',sparse=False)),
                       ('poly',PolynomialFeatures(degree = 2,interaction_only=True,include
_bias=False))])
pipe_cont_1 = Pipeline([('poly',PolynomialFeatures(degree=2, interaction_only=True,includ
```

```
e_bias=False)),
                      ('scaler',StandardScaler())])
cf_1 = ColumnTransformer([('cat',pipe_cat_1,cat_var),
                          ('cont',pipe_cont_1,cont_var)])

main_pipe_ridge_1 = Pipeline([('col_transformer',cf_1),
                       ('ridge',Ridge(fit_intercept=True))])
```

In [7]:

```
params_ridge_1 = {'ridge__alpha':np.logspace(-1,2,20)}

ridge_cv_1 = GridSearchCV(main_pipe_ridge_1,params_ridge_1,cv=5)
```

In [8]:

```
ridge_cv_1.fit(X_train,y_train)
```

Out[8]:

```
GridSearchCV(cv=5,
             estimator=Pipeline(steps=[('col_transformer',
                                        ColumnTransformer(transformers=[('cat',
                                                                         Pipeline(steps
=[('onehot',

OneHotEncoder(handle_unknown='ignore',

sparse=False)),

('poly',

PolynomialFeatures(include_bias=False,

interaction_only=True))]),
                                                                         ['yearID',
                                                                          'lgID']),
                                                                        ('cont',
                                                                         Pipeline(steps
=[('poly',

PolynomialFeatures(include_bias=False,

interact...
                                                                         Index(['2B_bat
', 'BB_ptch', 'CS_bat', 'GIDP_bat', 'HR_bat', 'HR_ptch', 'H_bat',
       'H_ptch', 'SB_bat', 'SO_ptch'],
      dtype='object'))])),
                                       ('ridge', Ridge())]),
             param_grid={'ridge__alpha': array([  0.1       ,   0.14384499,   0.20691381
,   0.29763514,
         0.42813324,   0.61584821,   0.88586679,   1.27427499,
         1.83298071,   2.6366509 ,   3.79269019,   5.45559478,
         7.8475997 ,  11.28837892,  16.23776739,  23.35721469,
        33.59818286,  48.32930239,  69.51927962, 100.        ])})
```

In [9]:

```
ridge_cv_1.best_params_
```

Out[9]:

```
{'ridge__alpha': 33.59818286283781}
```

In [10]:

```
ridge_cv_1.score(X_test,y_test)
```

Out[10]:

```
0.7968790227442946
```

In [11]:

```
main_pipe_rf_1 = Pipeline([('col_transformer',cf_1),
                           ('rf',RandomForestRegressor())])
```

In [12]:

```
params_rf_1 = {'rf__max_depth' : [1,2,3,4,5],
               'rf__min_samples_split': [2,3,4,5,6,7,8]}
```

In [13]:

```
rf_cv_1 = GridSearchCV(main_pipe_rf_1,params_rf_1,cv=5)
```

In [14]:

```
rf_cv_1.fit(X_train,y_train)
```

Out[14]:

```
GridSearchCV(cv=5,
             estimator=Pipeline(steps=[('col_transformer',
                                        ColumnTransformer(transformers=[('cat',
                                                                         Pipeline(steps
=[('onehot',

OneHotEncoder(handle_unknown='ignore',

sparse=False)),

('poly',

PolynomialFeatures(include_bias=False,

interaction_only=True))]),
                                                                         ['yearID',
                                                                          'lgID']),
                                                                        ('cont',
                                                                         Pipeline(steps
=[('poly',

PolynomialFeatures(include_bias=False,

interaction_only=True)),

('scaler',

StandardScaler())]),
                                                                         Index(['2B_bat
', 'BB_ptch', 'CS_bat', 'GIDP_bat', 'HR_bat', 'HR_ptch', 'H_bat',
       'H_ptch', 'SB_bat', 'SO_ptch'],
      dtype='object'))])),
                                       ('rf', RandomForestRegressor())]),
             param_grid={'rf__max_depth': [1, 2, 3, 4, 5],
                         'rf__min_samples_split': [2, 3, 4, 5, 6, 7, 8]})
```

In [15]:

```
rf_cv_1.score(X_test,y_test)
```

Out[15]:

```
0.7209534277724245
```

Bonus Question

In [16]:

```
pipe_cat = Pipeline([('onehot',OneHotEncoder(handle_unknown = 'ignore',sparse=False)),
                     ('poly',PolynomialFeatures(interaction_only=True,include_bias=False
```

```
))])
pipe_cont = Pipeline([('poly',PolynomialFeatures(interaction_only=True,include_bias=False
)),
                      ('scaler',StandardScaler())])
cf = ColumnTransformer([('cat',pipe_cat,cat_var),
                        ('cont',pipe_cont,cont_var)])

main_pipe_ridge = Pipeline([('col_transformer',cf),
                        ('ridge',Ridge(fit_intercept=True))])
```

In [17]:

```
params_ridge = {'ridge__alpha':np.logspace(-1,2,20),'col_transformer__cont__poly__degree
':[1,2,3,4,5]}

ridge_cv = GridSearchCV(main_pipe_ridge,params_ridge,cv=5)
```

In [18]:

```
ridge_cv.fit(X_train,y_train)
```

Out[18]:

```
GridSearchCV(cv=5,
             estimator=Pipeline(steps=[('col_transformer',
                                        ColumnTransformer(transformers=[('cat',
                                                                         Pipeline(steps
=[('onehot',

OneHotEncoder(handle_unknown='ignore',

sparse=False)),

('poly',

PolynomialFeatures(include_bias=False,

interaction_only=True))]),
                                                                         ['yearID',
                                                                          'lgID']),
                                                                        ('cont',
                                                                         Pipeline(steps
=[('poly',

PolynomialFeatures(include_bias=False,

interact...
       'H_ptch', 'SB_bat', 'SO_ptch'],
      dtype='object'))])),
                                        ('ridge', Ridge())]),
             param_grid={'col_transformer__cont__poly__degree': [1, 2, 3, 4, 5],
                         'ridge__alpha': array([  0.1       ,   0.14384499,   0.20691381
,   0.29763514,
         0.42813324,   0.61584821,   0.88586679,   1.27427499,
         1.83298071,   2.6366509 ,   3.79269019,   5.45559478,
         7.8475997 ,  11.28837892,  16.23776739,  23.35721469,
        33.59818286,  48.32930239,  69.51927962, 100.        ])})
```

In [19]:

```
ridge_cv.best_params_
```

Out[19]:

```
{'col_transformer__cont__poly__degree': 2, 'ridge__alpha': 33.59818286283781}
```

In [20]:

```
ridge_cv.score(X_test,y_test)
```

Out[20]:

```
0.7968790227442946
```

In [21]:

```
main_pipe_rf = Pipeline([('col_transformer',cf),
                         ('rf',RandomForestRegressor())])
```

In [22]:

```
params_rf = {'rf__max_depth' : [1,2,3,4,5],
             'rf__min_samples_split': [2,3,4,5,6,7,8]}
```

In [23]:

```
rf_cv = GridSearchCV(main_pipe_rf,params_rf,cv=5)
```

In [24]:

```
rf_cv.fit(X_train,y_train)
```

Out[24]:

```
GridSearchCV(cv=5,
             estimator=Pipeline(steps=[('col_transformer',
                                        ColumnTransformer(transformers=[('cat',
                                                                         Pipeline(steps
=[('onehot',

OneHotEncoder(handle_unknown='ignore',

sparse=False)),

('poly',

PolynomialFeatures(include_bias=False,

interaction_only=True))]),
                                                                         ['yearID',
                                                                          'lgID']),
                                                                        ('cont',
                                                                         Pipeline(steps
=[('poly',

PolynomialFeatures(include_bias=False,

interaction_only=True)),

('scaler',

StandardScaler())]),
                                                                         Index(['2B_bat
', 'BB_ptch', 'CS_bat', 'GIDP_bat', 'HR_bat', 'HR_ptch', 'H_bat',
       'H_ptch', 'SB_bat', 'SO_ptch'],
      dtype='object'))])),
                                       ('rf', RandomForestRegressor())]),
             param_grid={'rf__max_depth': [1, 2, 3, 4, 5],
                         'rf__min_samples_split': [2, 3, 4, 5, 6, 7, 8]})
```

In [25]:

```
rf_cv.score(X_test,y_test)
```

Out[25]:

```
0.7159170715976091
```