if_regression

May 4, 2020

[30]:
```
import pandas as pd
import matplotlib.pyplot as plt
from mlxtend.plotting import scatterplotmatrix
from mlxtend.plotting import heatmap
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score
from sklearn.linear_model import Ridge
from sklearn.linear_model import Lasso
from sklearn.linear_model import ElasticNet
from sklearn.ensemble import RandomForestRegressor
```

[2]:
```
data = pd.read_csv('../train.csv')
```

[3]:
```
data.head()
```

[3]:

	blueFirstBlood	blueKills	blueDeaths	blueGoldDiff	blueExperienceDiff	blueWardsPlacedDiff	blueWardsDestroyedDiff	blueAvgLevelDiff	blueAssistsDiff	blueTotalMinionsKilledDiff
0	0	5	3	976	1599	-21	2	0.4	1	10
1	1	5	4	780	523	-16	0	-0.2	0	0
2	0	6	14	-4443	-4140	1	0	-1.0	-5	-27
3	0	4	7	-1903	-584	-25	-1	0.0		
4	0	5	10	-3731	-1458	10	1	-0.6		

```
3               -8                          -10
4               -3                          -25

   blueTotalJungleMinionsKilledDiff  blueEliteMonstersDiff  blueDragonsDiff  \
0                                19                      1                1
1                                12                      2                1
2                               -20                     -1               -1
3                               -17                     -1               -1
4                                -5                     -1               -1

   blueHeraldsDiff  blueTowersDestroyedDiff  blueWins
0                0                        0         1
1                0                        0         1
2                0                        0         1
3                0                        0         0
4                0                        0         1
```

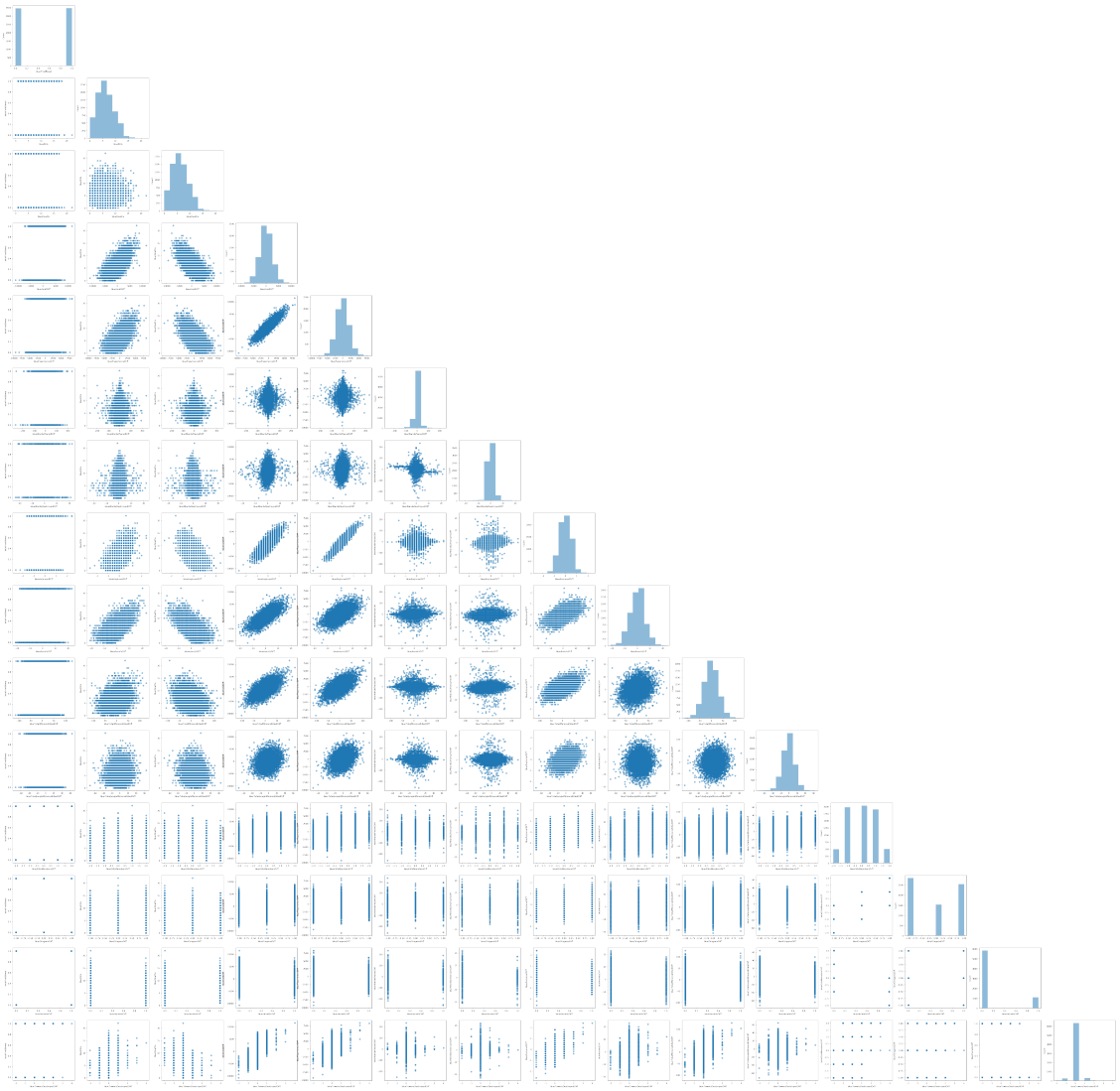
[4]:
```
data = data.iloc[:,:-1]
```

[5]:
```
data
```

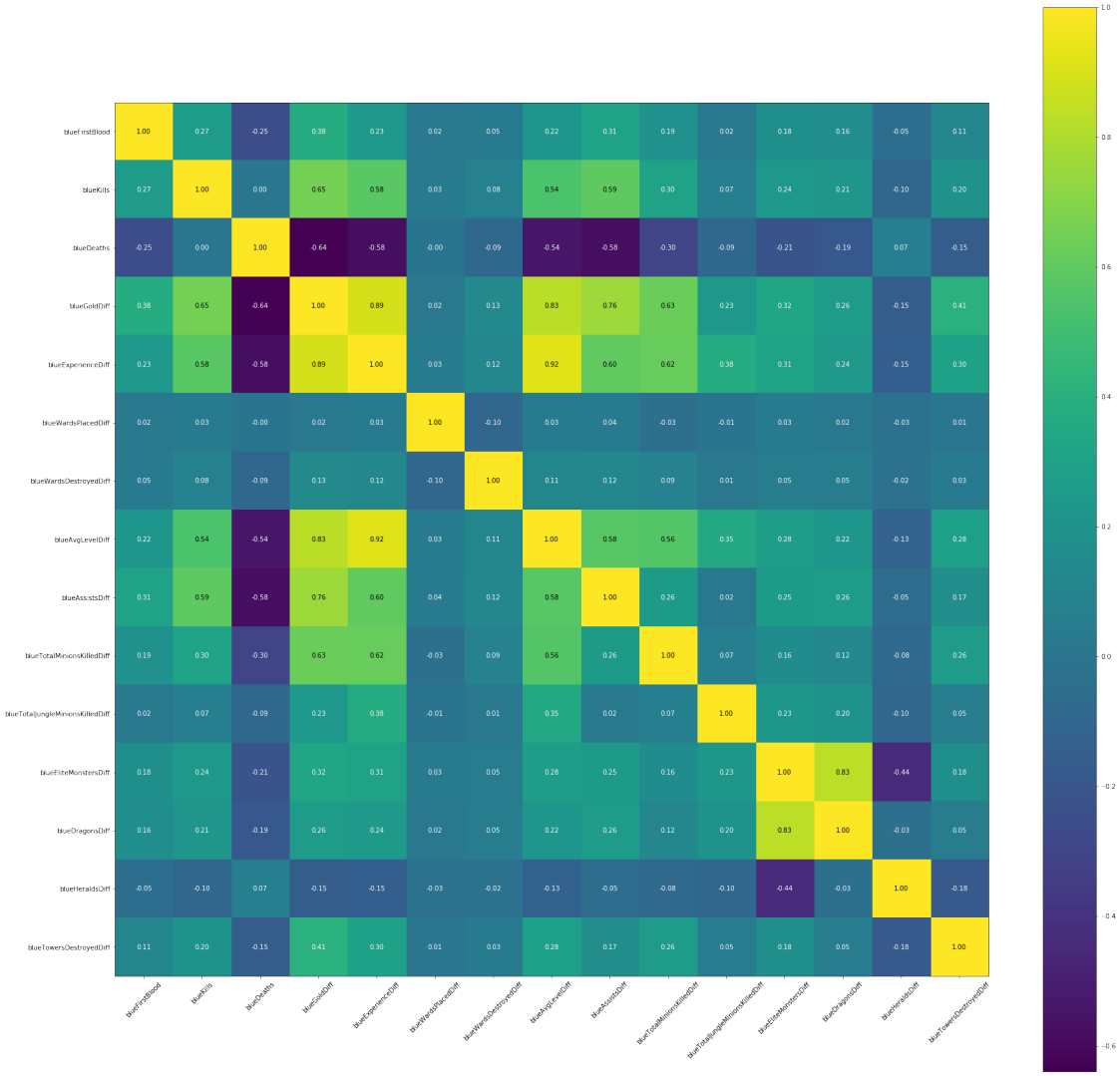
[5]:
```
      blueFirstBlood  blueKills  blueDeaths  blueGoldDiff  blueExperienceDiff  \
0                  0          5           3           976                1599
1                  1          5           4           780                 523
2                  0          6          14         -4443               -4140
3                  0          4           7         -1903                -584
4                  0          5          10         -3731               -1458
…                ...        ...         ...           ...                 ...
6910               0         10           4          3301                3347
6911               0          6           5           503                -194
6912               0          1           2            99                -174
6913               0          3          11         -4452               -3726
6914               1         11           3          4275                2943

      blueWardsPlacedDiff  blueWardsDestroyedDiff  blueAvgLevelDiff  \
0                     -21                       2               0.4
1                     -16                       0              -0.2
2                       1                       0              -1.0
3                     -25                      -1               0.0
4                      10                       1              -0.6
…                     ...                     ...               ...
6910                    1                       0               1.0
6911                   34                      -1              -0.4
6912                    3                       1               0.2
6913                   26                      -1              -0.6
6914                   20                       3               1.0
```

```
      blueAssistsDiff  blueTotalMinionsKilledDiff  \
0                   1                          10
1                   0                           0
2                  -5                         -27
3                  -8                         -10
4                  -3                         -25
…                 …                           …
6910                0                          36
6911                2                          19
6912               -4                          34
6913               -4                         -48
6914                8                          32

      blueTotalJungleMinionsKilledDiff  blueEliteMonstersDiff  \
0                                   19                      1
1                                   12                      2
2                                  -20                     -1
3                                  -17                     -1
4                                   -5                     -1
…                                  …                       …
6910                                15                     -1
6911                               -44                     -2
6912                                -4                      1
6913                               -13                      0
6914                                -1                      1

      blueDragonsDiff  blueHeraldsDiff  blueTowersDestroyedDiff
0                   1                0                        0
1                   1                0                        0
2                  -1                0                        0
3                  -1                0                        0
4                  -1                0                        0
…                 …                 …                        …
6910               -1                0                        0
6911               -1                1                        0
6912                1                0                        0
6913               -1                0                        0
6914                1                0                        0

[6915 rows x 15 columns]
```

[6]:
```
scatterplotmatrix(data.values, figsize=(90,90),names=data.columns, alpha=0.5)
plt.show()
```



[7]:
```
cm = np.corrcoef(data.values.T)
hm = heatmap(cm, row_names=data.columns, column_names=data.columns,␣
 ↪figsize=(30,30))
plt.show()
```



[8]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6915 entries, 0 to 6914
Data columns (total 15 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   blueFirstBlood                6915 non-null   int64
 1   blueKills                     6915 non-null   int64
 2   blueDeaths                    6915 non-null   int64
 3   blueGoldDiff                  6915 non-null   int64
 4   blueExperienceDiff            6915 non-null   int64
 5   blueWardsPlacedDiff           6915 non-null   int64
 6   blueWardsDestroyedDiff        6915 non-null   int64
```

```
 7   blueAvgLevelDiff                 6915 non-null   float64
 8   blueAssistsDiff                  6915 non-null   int64
 9   blueTotalMinionsKilledDiff       6915 non-null   int64
 10  blueTotalJungleMinionsKilledDiff 6915 non-null   int64
 11  blueEliteMonstersDiff            6915 non-null   int64
 12  blueDragonsDiff                  6915 non-null   int64
 13  blueHeraldsDiff                  6915 non-null   int64
 14  blueTowersDestroyedDiff          6915 non-null   int64
dtypes: float64(1), int64(14)
memory usage: 810.5 KB
```

[72]:
```
X = data[data.columns.difference(['blueGoldDiff'])]
y = data['blueGoldDiff']
```

[73]:
```
X.shape
```

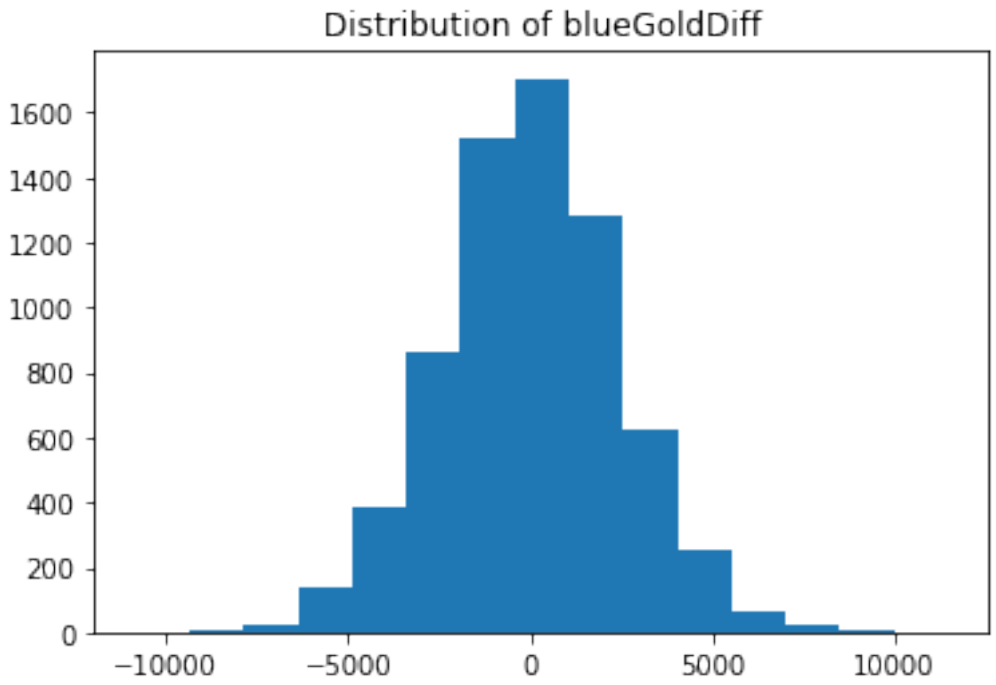
[73]:
```
(6915, 14)
```

[74]:
```
y.describe()
```

[74]:
```
count     6915.000000
mean        19.731743
std       2451.507571
min     -10830.000000
25%      -1558.500000
50%          2.000000
75%       1605.000000
max      11467.000000
Name: blueGoldDiff, dtype: float64
```

[75]:
```
plt.hist(y,bins=15)
plt.title('Distribution of blueGoldDiff')
```

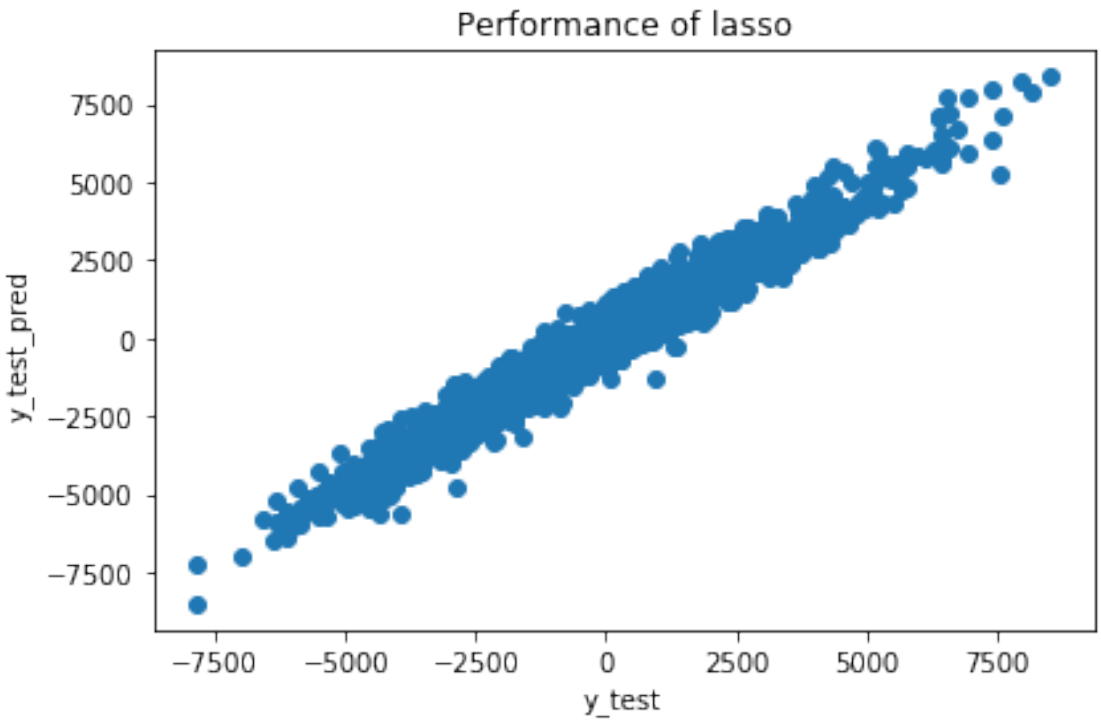
[75]:
```
Text(0.5, 1.0, 'Distribution of blueGoldDiff')
```



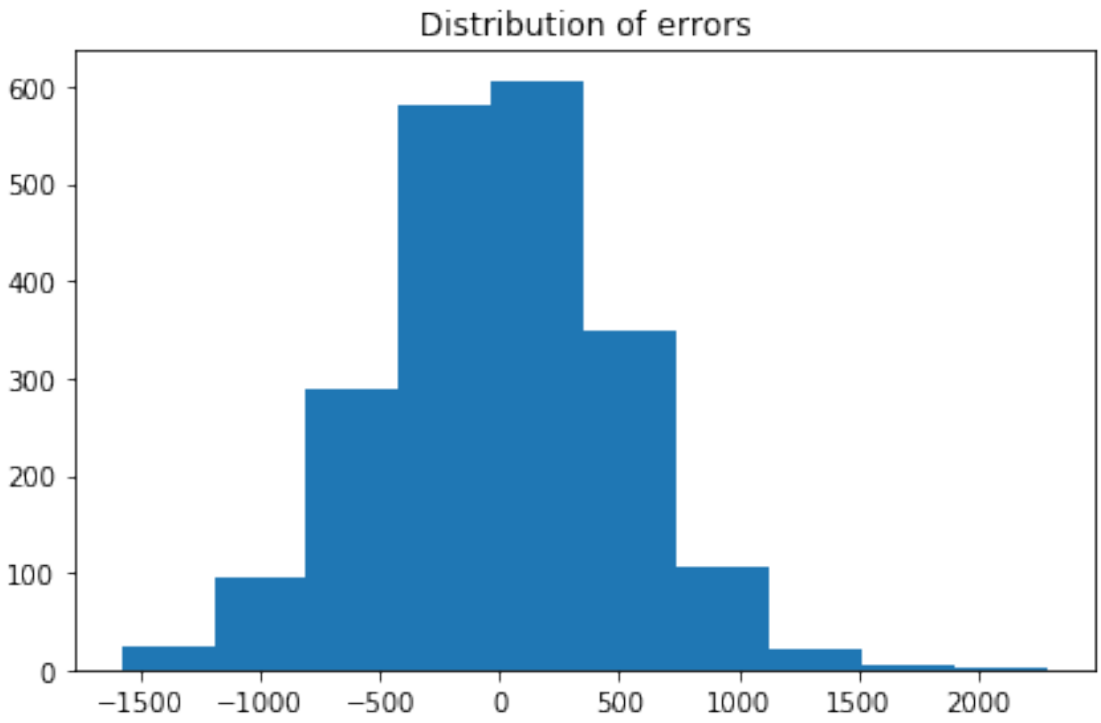
1 Lasso

[76]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,␣
↪random_state=42)
lasso = Lasso(random_state=42)
lasso.fit(X_train,y_train)
y_test_pred = lasso.predict(X_test)
y_train_pred = lasso.predict(X_train)
mse = mean_squared_error(y_test, y_test_pred)
print(mse)
plt.scatter(y_test, y_test_pred)
plt.title('Performance of lasso')
plt.xlabel('y_test')
plt.ylabel('y_test_pred')
plt.tight_layout()
```

```
251108.76923395693
```



[77]:
```
plt.hist(y_test-y_test_pred)
plt.title('Distribution of errors')
plt.tight_layout()
```



[84]:
```
X = data[data.columns.difference(['blueGoldDiff'])]
y = data['blueGoldDiff']

sc = StandardScaler()
X_sc = sc.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X_sc, y, test_size=0.3,␣
↪random_state=42)
lasso = Lasso(random_state=42)
lasso.fit(X_train,y_train)
y_test_pred = lasso.predict(X_test)
y_train_pred = lasso.predict(X_train)
mse = mean_squared_error(y_test, y_test_pred)
print(mse)
```

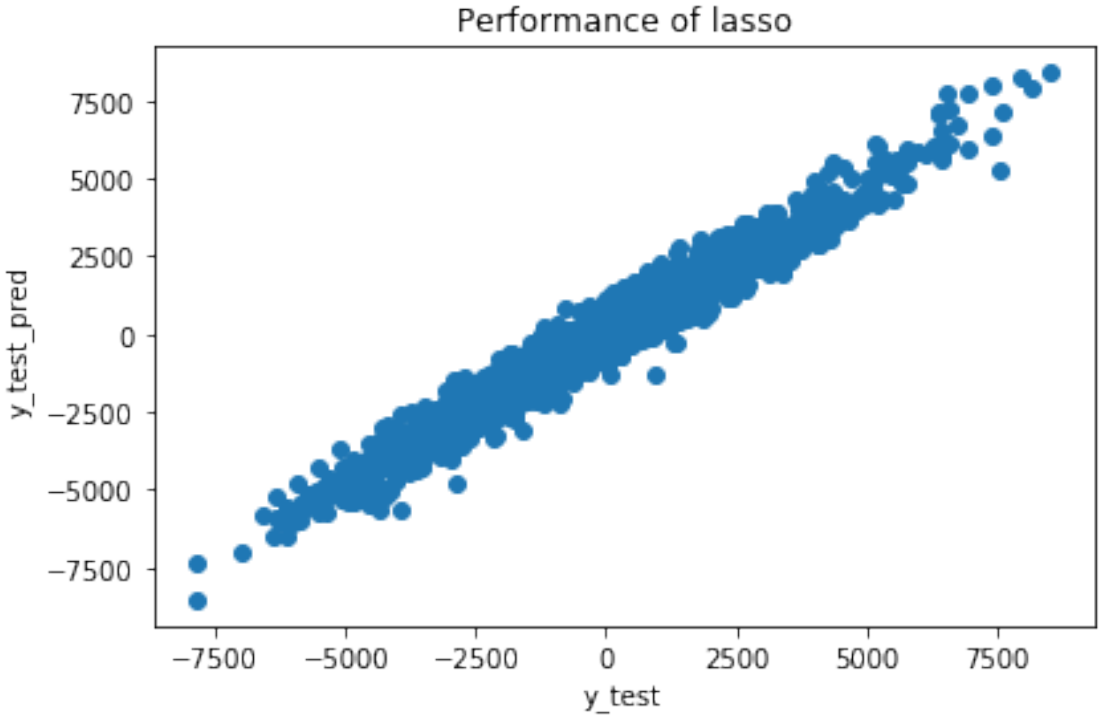
```
250919.78957446507
```

2 Ridge

[86]:
```
X = data[data.columns.difference(['blueGoldDiff'])]
y = data['blueGoldDiff']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,␣
↪random_state=42)
ridge = Ridge(random_state=42)
ridge.fit(X_train,y_train)
y_test_pred = ridge.predict(X_test)
y_train_pred = ridge.predict(X_train)
mse = mean_squared_error(y_test, y_test_pred)
print(mse)
plt.scatter(y_test, y_test_pred)
plt.title('Performance of lasso')
plt.xlabel('y_test')
plt.ylabel('y_test_pred')
plt.tight_layout()
```

```
250913.55171901098
```



[88]:
```
X = data[data.columns.difference(['blueGoldDiff'])]
y = data['blueGoldDiff']

sc = StandardScaler()
X_sc = sc.fit_transform(X)
```

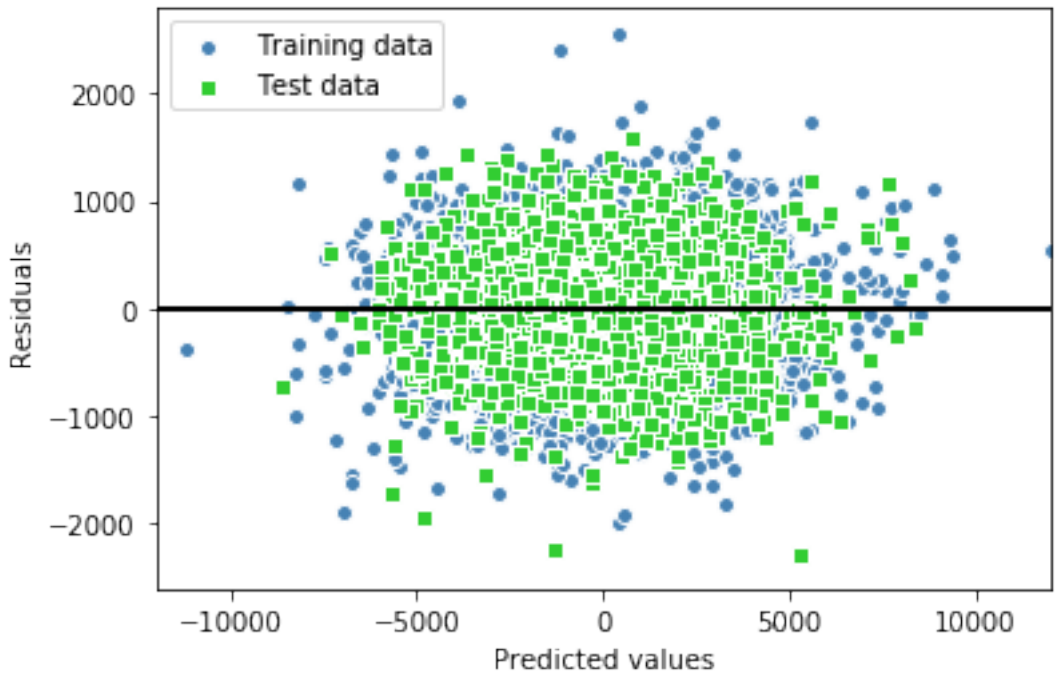
```
X_train, X_test, y_train, y_test = train_test_split(X_sc, y, test_size=0.3,␣
 ↪random_state=42)
ridge = Ridge(random_state=42)
ridge.fit(X_train,y_train)
y_test_pred = ridge.predict(X_test)
y_train_pred = ridge.predict(X_train)
mse = mean_squared_error(y_test, y_test_pred)
print(mse)
```

```
250900.85923226248
```

3 Reression

[89]:
```
X = data[data.columns.difference(['blueGoldDiff'])]
y = data['blueGoldDiff']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,␣
 ↪random_state=42)
lr = LinearRegression()
lr.fit(X_train, y_train)
y_train_pred = lr.predict(X_train)
y_test_pred = lr.predict(X_test)
```

[90]:
```
plt.scatter(y_train_pred, y_train_pred - y_train,c='steelblue', marker='o',␣
 ↪edgecolor='white',label='Training data')
plt.scatter(y_test_pred, y_test_pred - y_test,c='limegreen', marker='s',␣
 ↪edgecolor='white',label='Test data')
plt.xlabel('Predicted values')
plt.ylabel('Residuals')
plt.legend(loc='upper left')
plt.hlines(y=0, xmin=-12000, xmax=12000, color='black', lw=2)
plt.xlim([-12000, 12000])
plt.show()
```



[91]:
```
print('MSE train: %.3f, test: %.3f' % (mean_squared_error(y_train,␣
↪y_train_pred),
                                       mean_squared_error(y_test, y_test_pred)))
```

```
MSE train: 260335.294, test: 250903.936
```

[92]:
```
print('R^2 train: %.3f, test: %.3f' %
        (r2_score(y_train, y_train_pred),
        r2_score(y_test, y_test_pred)))
```

```
R^2 train: 0.957, test: 0.957
```

[ ]:

[ ]: