Regression_problem

May 4, 2016

```
In [4]: import pandas as pd
        import numpy as np
        from sklearn import cross_validation
        from sklearn import svm
        from sklearn.tree import DecisionTreeRegressor
        from sklearn.ensemble import RandomForestRegressor
        from sklearn.linear_model import LinearRegression
        from sklearn.linear_model import Ridge
        from sklearn.linear_model import Lasso
        from sklearn.neighbors import KNeighborsRegressor
        from sklearn.metrics import mean_squared_error
```

```
In [7]: df = pd.read_csv('housing.csv',sep=',',header=None)
        #shuffle the data
        df = df.iloc[np.random.permutation(len(df))]
        X= df[df.columns[:-1]].values
        Y = df[df.columns[-1]].values

        cv = 10
        print 'linear regression'
        lin = LinearRegression()
        scores = cross_validation.cross_val_score(lin, X, Y, cv=cv)
        print("mean R2: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
        predicted = cross_validation.cross_val_predict(lin, X,Y, cv=cv)
        print 'MSE:',mean_squared_error(Y,predicted)

        print 'ridge regression'
        ridge = Ridge(alpha=1.0)
        scores = cross_validation.cross_val_score(ridge, X, Y, cv=cv)
        print("mean R2: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
        predicted = cross_validation.cross_val_predict(ridge, X,Y, cv=cv)
        print 'MSE:',mean_squared_error(Y,predicted)

        print 'lasso regression'
        lasso = Lasso(alpha=0.1)
        scores = cross_validation.cross_val_score(lasso, X, Y, cv=cv)
        print("mean R2: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
        predicted = cross_validation.cross_val_predict(lasso, X,Y, cv=cv)
        print 'MSE:',mean_squared_error(Y,predicted)

        print 'decision tree regression'
        tree = DecisionTreeRegressor(random_state=0)
        scores = cross_validation.cross_val_score(tree, X, Y, cv=cv)
        print("mean R2: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
        predicted = cross_validation.cross_val_predict(tree, X,Y, cv=cv)
        print 'MSE:',mean_squared_error(Y,predicted)

        print 'random forest regression'
        forest = RandomForestRegressor(n_estimators=50, max_depth=None,min_samples_split=1, random_stat
        scores = cross_validation.cross_val_score(forest, X, Y, cv=cv)
        print("mean R2: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
        predicted = cross_validation.cross_val_predict(forest, X,Y, cv=cv)
        print 'MSE:',mean_squared_error(Y,predicted)

        #svm
        print 'linear support vector machine'
        svm_lin = svm.SVR(epsilon=0.2,kernel='linear',C=1)
        scores = cross_validation.cross_val_score(svm_lin, X, Y, cv=cv)
        print("mean R2: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
        predicted = cross_validation.cross_val_predict(svm_lin, X,Y, cv=cv)
        print 'MSE:',mean_squared_error(Y,predicted)

        print 'support vector machine rbf'
        clf = svm.SVR(epsilon=0.2,kernel='rbf',C=1.)
        scores = cross_validation.cross_val_score(clf, X, Y, cv=cv)
        print("mean R2: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
        predicted = cross_validation.cross_val_predict(clf, X,Y, cv=cv)
        print 'MSE:',mean_squared_error(Y,predicted)

        print 'knn'
        knn = KNeighborsRegressor()
        scores = cross_validation.cross_val_score(knn, X, Y, cv=cv)
        print("mean R2: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
        predicted = cross_validation.cross_val_predict(knn, X,Y, cv=cv)
        print 'MSE:',mean_squared_error(Y,predicted)
```

```
linear regression
mean R2: 0.72 (+/- 0.15)
MSE: 23.5515499366
ridge regression
mean R2: 0.72 (+/- 0.16)
MSE: 23.7397585761
lasso regression
mean R2: 0.71 (+/- 0.17)
MSE: 24.734860679
decision tree regression
mean R2: 0.75 (+/- 0.24)
MSE: 19.8023913043
random forest regression
mean R2: 0.87 (+/- 0.12)
MSE: 10.9910313913
linear support vector machine
mean R2: 0.70 (+/- 0.25)
MSE: 25.833801836
support vector machine rbf
mean R2: -0.01 (+/- 0.11)
MSE: 83.8283880541
knn
```

```
mean R2: 0.54 (+/- 0.23)
MSE: 37.8792632411
```

```
In [9]: from sklearn.feature_selection import RFE
        best_features=4
        print 'feature selection on linear regression'
        rfe_lin = RFE(lin,best_features).fit(X,Y)
        mask = np.array(rfe_lin.support_)
        scores = cross_validation.cross_val_score(lin, X[:,mask], Y, cv=cv)
        print("R2: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
        predicted = cross_validation.cross_val_predict(lin, X[:,mask],Y, cv=cv)
        print 'MSE:',mean_squared_error(Y,predicted)

        print 'feature selection ridge regression'
        rfe_ridge = RFE(ridge,best_features).fit(X,Y)
        mask = np.array(rfe_ridge.support_)
        scores = cross_validation.cross_val_score(ridge, X[:,mask], Y, cv=cv)
        print("R2: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
        predicted = cross_validation.cross_val_predict(ridge, X[:,mask],Y, cv=cv)
        print 'MSE:',mean_squared_error(Y,predicted)

        print 'feature selection on lasso regression'
        rfe_lasso = RFE(lasso,best_features).fit(X,Y)
        mask = np.array(rfe_lasso.support_)
        scores = cross_validation.cross_val_score(lasso, X[:,mask], Y, cv=cv)
        print("R2: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
        predicted = cross_validation.cross_val_predict(lasso, X[:,mask],Y, cv=cv)
        print 'MSE:',mean_squared_error(Y,predicted)

        print 'feature selection on decision tree'
        rfe_tree = RFE(tree,best_features).fit(X,Y)
        mask = np.array(rfe_tree.support_)
        scores = cross_validation.cross_val_score(tree, X[:,mask], Y, cv=cv)
        print("R2: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
        predicted = cross_validation.cross_val_predict(tree, X[:,mask],Y, cv=cv)
        print 'MSE:',mean_squared_error(Y,predicted)

        print 'feature selection on random forest'
        rfe_forest = RFE(forest,best_features).fit(X,Y)
        mask = np.array(rfe_forest.support_)
        scores = cross_validation.cross_val_score(forest, X[:,mask], Y, cv=cv)
        print("R2: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
        predicted = cross_validation.cross_val_predict(forest, X[:,mask],Y, cv=cv)
        print 'MSE:',mean_squared_error(Y,predicted)

        print 'feature selection on linear support vector machine'
        rfe_svm = RFE(svm_lin,best_features).fit(X,Y)
        mask = np.array(rfe_svm.support_)
        scores = cross_validation.cross_val_score(svm_lin, X[:,mask], Y, cv=cv)
        print("R2: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
        predicted = cross_validation.cross_val_predict(svm_lin, X,Y, cv=cv)
        print 'MSE:',mean_squared_error(Y,predicted)

        print 'feature selection on knn'
```

```
rfe_knn = RFE(knn,best_features).fit(X,Y)
mask = np.array(knn.support_)
scores = cross_validation.cross_val_score(knn, X[:,mask], Y, cv=cv)
print("R2: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
predicted = cross_validation.cross_val_predict(knn, X,Y, cv=cv)
print ’MSE:’,mean_squared_error(Y,predicted)
```

```
feature selection on linear regression
R2: 0.61 (+/- 0.31)
MSE: 33.182126206
feature selection ridge regression
R2: 0.61 (+/- 0.32)
MSE: 33.2543979822
feature selection on lasso regression
R2: 0.68 (+/- 0.20)
MSE: 27.4174043724
feature selection on decision tree
R2: 0.70 (+/- 0.35)
MSE: 24.1185968379
feature selection on random forest
R2: 0.84 (+/- 0.14)
MSE: 13.6755712332
feature selection on linear svm
R2: 0.60 (+/- 0.33)
MSE: 25.833801836
feature selection on knn
```

```
        ---------------------------------------------------------------------------

        RuntimeError                              Traceback (most recent call last)

        <ipython-input-9-0445c028402d> in <module>()
         50
         51 print ’feature selection on knn’
    ---> 52 rfe_knn = RFE(knn,best_features).fit(X,Y)
         53 mask = np.array(knn.support_)
         54 scores = cross_validation.cross_val_score(knn, X[:,mask], Y, cv=cv)


        /lib/python2.7/site-packages/sklearn/feature_selection/rfe.pyc in fit(self, X, y)
        129                 The target values.
        130             """
    --> 131             return self._fit(X, y)
        132
        133         def _fit(self, X, y, step_score=None):


        /lib/python2.7/site-packages/sklearn/feature_selection/rfe.pyc in _fit(self, X, y, step_score)
        180                     coefs = estimator.feature_importances_
        181                 else:
    --> 182                     raise RuntimeError(’The classifier does not expose ’
        183                                        ’"coef_" or "feature_importances_" ’
        184                                        ’attributes’)
```

```
RuntimeError: The classifier does not expose "coef_" or "feature_importances_" attributes
```

```
In [ ]:
```