Effectiveness of ensemble machine learning over the conventional multivariable linear regression models

Yoshiyasu Takefuji
Faculty of Environment and information studies
Keio University
5322 Endo, Fujisawa, 2520882 JAPAN
takefuji@sfc.keio.ac.jp

Koichiro Shoji
Science Park, Corp
3-1649-2, Iriya Zama-shi, 252-0024 Japan
shoji@sciencepark.co.jp

***Abstract*—This paper demonstrates the effectiveness of ensemble machine learning algorithms over the conventional multivariable linear regression models including Ordinary Least Squares, Robust Linear Model, and Lasso Model. The ensemble machine learning algorithms include Adaboost, Random-Forest, Bagging, Extremely Randomized Trees, Gradient Boosting, and Extra Trees Regressor. With the progress of open sources, a variety of algorithms are available and they can be easily compared by using open source Python libraries from the viewpoint of prediction accuracies using R-squared.**

Keywords—big data, ensemble machine learning, OLS, RLM, Lasso

I. INTRODUCTION

A variety of algorithms have been proposed for predicting the correlations between input and output. The conventional multivariable linear regression models have been used for predicting output, for example, sales, with several parameters (input) where the examined models include an Ordinary Least Squares (OLS), a Robust Linear Model (RLM), and a Lasso model from open sources (statsmodels) [1]. With the rapid progress of open source machine learning (sklearn:scikit-learn) [2], ensemble machine learning algorithms including Adaboost, Random-Forest, Bagging, Extremely Randomized Trees, Gradient Boosting, and Extra Trees Regressor can be applied to the input/output correlation problems.

This paper demonstrates the effectiveness of ensemble machine learning algorithms by comparing the conventional multivariable linear regression models using R-squared. In order to evaluate the quality of the algorithms, R-squared is used in this paper to measure goodness-of-fit in regression.

A given problem in this paper is to predict the ice-cream sales by the temperature and the number of pedestrians in the street. The more number of pedestrians, the ice-cream sales increase. The temperature is one of important factors for ice-cream sales. Based on the latest research [3], the ice-cream is the most seasonable food for summer and there is a strong correlation between the ice-cream sales and the temperature. At 18 degrees Celsius, sales soar.

Without such pre-knowledge about the ice-cream sales, the ensemble machine learning models outperform the existing multivariable linear regression models. The data of the ice-cream sales, the number of pedestrians, the temperature are downloadable from:

http://xica-inc.com/adelie/sample/data/ice.zip

or

http://web.sfc.keio.ac.jp/~takefuji/ice.csv

II. MULTIVARIABLE LINEAR REGRESSION MODELS

A. Ordinary Least Squares(OLS)

Ordinary Least Squares (OLS) model is a classical multivariable linear regression model. OLS is a statistical method which attempts to find the function which most closely approximates the data, so called a best fit. The Least Squares method is used to fit a straight line through a set of data-points, so that the sum of the squared vertical distances from the actual data-points is minimized. Open source Python library, "statsmodels" is used in this paper.

Downloaded ice.csv data is composed of the ice sales data['ice'], data['temp'] for temperature, and data['street'] for the number of pedestrians. The following important source codes (imported library name, x:input, y:output, p:predicted output) describe the OLS regression Python program where "tem" variable and "st" variable indicate the coefficient of the temperature and that of the number of pedestrians respectively. The pandas library is used for data manipulations.

```
import statsmodels.api, pandas
data=pandas.read_csv('ice.csv')
x=data[['temp','street']]
x= statsmodels.api.add_constant(x)
y=data['ice']
est=statsmodels.api.OLS(y,x).fit()
const,tem,st=est.params
p= tem*data['temp']+st*data['street']+const
```

Fig.1 shows the result of real ice sales (y: dotted line) and OLS predicted sales (p: solid line). The vertical axis indicates the ice-cream sales (Japanese Yen) while the horizontal axis means experimented 31 days for summer. Computed R-squared is 0.45 which shows no-good-fitting. R-squared

indicates a measure of goodness-of-fit in regressions within a range of 0 to 1 where the higher, the better.

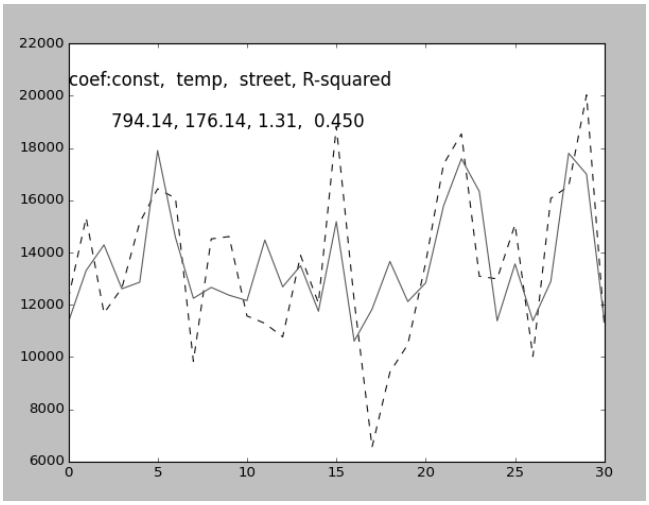

Fig.1 Ice sales (dotted line) and OLS predicted (solid line): vertical axis (sales), horizontal axis (30 days)

B. RLM and Lasso model

In order to implement the RLM algorithm using statsmodels library, the followings are important source codes (imported library name, clf:classfier, p:predicted output) of the Python program:

```
import statsmodels.api as sm
import pandas
est= sm.RLM(y,x,M=sm.robust.norms.HuberT()).fit()
const,tem,st=est.params
p= tem*data['temp']+st*data['street']+const
```

In Lasso model from sklearn library, the followings are important codes of the Python program:

```
from sklearn import linear_model
clf=linear_model.Lasso()
clf.fit(x,y)
p=clf.predict(x)
```

The similar results as shown in Fig. 1 were obtained by RLM and Lasso algorithms respectively where R-squared is 0.45.

37 linear_model methods in the sklearn library are available and 10 methods were examined as shown in Table-1.

The maximum R-squared is 0.45 in examined linear_model methods.

Table-1 Experimented linear_model methods in sklearn

Linear_model methods	R-squared
linear_model.ARDRegression	0.437
linear_model.BayesianRidge	0.421
linear_model.ElasticNet	0.45
llinear_model.Lars	0.45
linear_model.Lasso	0.45
linear_model.LassoLars	0.45
linear_model.LinearRegression	0.45
linear_model.LogisticRegression	0.323
linear_model.OrthogonalMatchingPursuit	0.421
linear_model.Ridge	0.45

III. ENSEMBLE MACHINE LEARNING

11 ensemble machine learning algorithms have been proposed and implemented in open sources including sklearn library (scikit-learn) as shown in Table-2.

Table-2 Ensemble Methods in sklearn

Ensemble methods	details
ensemble.AdaBoostClassifier	An AdaBoost classifier
ensemble.AdaBoostRegressor	An AdaBoost regressor
ensemble.BaggingClassifier	A Bagging classifier
ensemble.BaggingRegressor	A Bagging regressor
ensemble.ExtraTreesClassifier	An extra-trees classifier
ensemble.ExtraTreesRegressor	An extra-trees regressor
ensemble.GradientBoostingClassifier	Gradient Boosting for classification
ensemble.GradientBoostingRegressor	Gradient Boosting for regression
ensemble.RandomForestClassifier	A random forest classifier
ensemble.RandomTreesEmbedding	An ensemble of totally random trees
ensemble.RandomForestRegressor	A random forest regressor

Adaboost with DecisionTree, Random-Forest, Extremely Randomized Trees, Bagging, Gradient Boosting, and Extra-Trees-Regressor were investigated in this paper [4].

Each ensemble machine learning method uses multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms.

In this Section, important codes [imported library name, clf:classifier, p:predicted output, clf.score(x,y): R-squared] of each ensemble machine learning are only shown by the followings:

Adaboost with DecisionTree :

```
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import AdaBoostRegressor
clf1=DecisionTreeRegressor(max_depth=4)
clf2=AdaBoostRegressor(DecisionTreeRegressor(max_depth=4),n_estimators=300,random_state=rng)
clf1.fit(x,y)
clf2.fit(x,y)
p1=clf1.predict(x)
p2=clf2.predict(x)
print clf1.score(x,y)
print clf2.score(x,y)
```

Random Forest:

```
from sklearn.ensemble import RandomForestRegressor
clf=RandomForestRegressor(n_estimators=200, min_samples_split=1)
clf.fit(x,y)
p=clf.predict(x)
print clf.score(x,y)
```

Extremely Randomized Tree:

```
from sklearn.ensemble import ExtraTreesClassifier
clf = ExtraTreesClassifier(n_estimators=100, max_depth=None,min_samples_split=1, random_state=0)
clf.fit(x,y)
p=clf.predict(x)
print clf.score(x,y)
```

Bagging with KNeighbors:

```
from sklearn.ensemble import BaggingClassifier
from sklearn.neighbors import KNeighborsClassifier
clf=BaggingClassifier(KNeighborsClassifier(), n_estimators=1000,max_samples=0.8, max_features=0.5)
clf.fit(x,y)
p=clf.predict(x)
print clf.score(x,y)
```

Gradient Boosting codes:

```
from sklearn.ensemble import GradientBoostingRegressor
clf=GradientBoostingRegressor(n_estimators=1000, learning_rate=1.2,max_depth=1, random_state=0)
clf.fit(x,y)
p=clf.predict(x)
print clf.score(x,y)
```

Extra-Trees-Regressor codes:

```
from sklearn.tree import ExtraTreeRegressor
clf=ExtraTreeRegressor()
clf.fit(x,y)
p=clf.predict(x)
print clf.score(x,y)
```

Fig. 2 shows the result of Gradient boosting over the real sales data where both lines are almost fitted with R-squared=0.985.

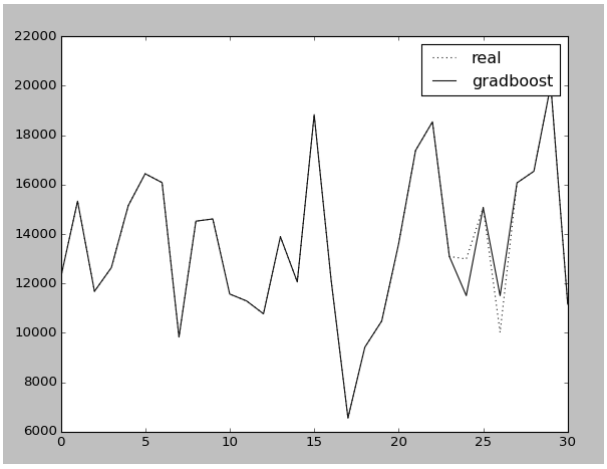

Fig. 2 Ice sales (dotted line), Gradient Boosting (solid line)

vertical axis (sales), horizontal axis (30 days)

Computed R-squared data of DecisionTree, Adaboost with DecisionTree, RandomForestRegressor, Bagging with KNeighborsClassifier, Extremely Randomized Tree, Gradient Boosting, and ExtraTreeRegressor are described respectively as shown in Table-3.

All ensemble machine learning algorithms can significantly improve the R-squared data over the conventional linear regression algorithms with R-squared=0.45. Table-3 shows that Gradient Boosting and ExtraTreeRegressor are the best algorithms among all algorithms with R-squared=0.985.

Table-3 R-squared of examined algorithms

Algorithms	R-squared
OLS, RLM, Lasso	0.45
DecisionTree	0.766
Adaboost with DecisionTree	0.959
RandomForestRegressor	0.864
RandomForestClassifier	0.968
Bagging with KNeighbors	0.935
ExtremelyRandomizedTree	0.968
GradientBoosting	0.985
ExtraTreeRegressor	0.985

A. Implemented Python program

The Python full program for Adaboost with DecisionTree is described in Fig. 4.

```
import pandas as pd
import numpy as np
import statsmodels.api as sm
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import AdaBoostRegressor
import matplotlib.pyplot as plt
data=pd.read_csv('ice.csv')
x=data[['temp','street']]
y=data['ice']
rng=np.random.RandomState(1)
clf1=DecisionTreeRegressor(max_depth=4)
clf2=AdaBoostRegressor(DecisionTreeRegressor(max_depth=4),n_estimators=300,random_state=rng)
clf1.fit(x,y)
clf2.fit(x,y)
p1=clf1.predict(x)
p2=clf2.predict(x)
print clf1.score(x,y)
print clf2.score(x,y)
t=np.arange(0.0,31.0)
plt.plot(t,data['ice'],'--b')
plt.plot(t,p1,':b')
plt.plot(t,p2,'-b')
plt.legend(('real','dtree','adaB'))
plt.show()
```

Fig. 4 adaboost.py

B. How to install statsmodels and sklearn libraries on Windows and Linux

This Section shows how to install statsmodels and sklearn Python libraries.

After installing Python on your system, run the following commands in the super user mode:

easily_install statsmodels

In order to install sklearn on Windows:

pip install -U scikit-learn

or

download *.whl from

http://www.lfd.uci.edu/~gohlke/pythonlibs/#scikit-learn

pip install xxx.whl

On Linux:

apt-get install build-essential python-dev python-setuptools python-numpy python-scipy libatlas-dev libatlas3gf-base

IV. FUTURE WORKS

There are a variety number of combinations using ensemble machine learning. We should further investigate what combinations will give us the best performance.

V. CONCLUSIONS

We have investigated the performance of ensemble machine learning algorithms over the conventional linear regressions. Ensemble machine learning algorithms including Gradient-Boosting and Extra-Tree-Regressor have generated

the best performance with R-squared=0.985 while the conventional linear regression algorithms have R-squared=0.45.

The ensemble machine learning can significantly improve R-squared, goodness-of-fit in regressions over the existing linear regression algorithms.

REFERENCES.

[1] http://statsmodels.sourceforge.net/

[2] http://scikit-learn.org/stable/

[3] http://www.unilever.co.uk/media-centre/pressreleases/2014/Perfect-temperature-for-ice-cream-sales-spike-revealed-and-its-lower-than-youd-think.aspx

[4] Yoshiyasu Takefuji, Introduction to IoT design and implementations using Open Sources (Ohmsha 2015).