



Ενότητα 5 – Supervised Learning *Linear Regression*

Μέθοδοι Μηχανικής Μάθησης στα Χρηματοοικονομικά

Αθανάσιος Σάκκας, Επ. Καθηγητής, ΟΠΑ

Γραμμική Παλινδρόμηση (Linear Regression)

- Η γραμμική παλινδρόμηση είναι ένα πολύ δημοφιλές εργαλείο, επειδή αφού κάνετε την υπόθεση ότι το μοντέλο είναι γραμμικό, δεν χρειάζεστε τεράστιο όγκο δεδομένων
- Στη μηχανική μάθηση αναφερόμαστε στον σταθερό όρο ως bias και στους συντελεστές ως weights

Υποθέστε n παρατηρήσεις και m χαρακτηριστικά. Το μοντέλο είναι το ακόλουθο

$$Y = a + b_1X_1 + b_2X_2 + \dots + b_mX_m + e$$

Η τυπική προσέγγιση είναι να επιλέξετε το a και το b_i για να ελαχιστοποιήσετε το μέσο τετραγωνικό σφάλμα (mean square error - mse). Μπορείτε να αξιολογήσετε την προγνωστική απόδοση του μοντέλου συγκρίνοντας το μέσο τετραγωνικό σφάλμα (mean square error - mse) στα test data και training data.

Αυτό μπορεί να γίνει αναλυτικά με χρήση πινάκων. Για καταστάσεις όπου απαιτείται ελάχιστο και δεν υπάρχει αναλυτική λύση, μπορεί να χρησιμοποιηθεί ο αλγόριθμος gradient descent.

Αλγόριθμος gradient descent

Ο στόχος είναι η ελαχιστοποίηση μιας συνάρτησης αλλάζοντας παραμέτρους.

Τα βήματα είναι τα εξής:

1. Επιλέξτε αρχική τιμή για τις παραμέτρους.
2. Βρείτε την πιο απότομη κλίση steepest slope : δηλαδή την κατεύθυνση στην οποία πρέπει να αλλάξει η παράμετρος για να μειωθεί η αντικειμενική συνάρτηση κατά το μεγαλύτερο ποσό.
3. Κάντε ένα βήμα κάτω προς την κατεύθυνση της πιο απότομη κλίση.
4. Επαναλάβετε τα βήματα 2 και 3.
5. Συνεχίστε μέχρι να φτάσετε στο τέλος.

Categorical features (κατηγορικά χαρακτηριστικά)

- Τα κατηγορικά χαρακτηριστικά είναι χαρακτηριστικά όπου υπάρχει ένας αριθμός μη αριθμητικών εναλλακτικών
- Μπορούμε να ορίσουμε μια Dummy Variable για κάθε εναλλακτική. Η μεταβλητή ισούται με 1 εάν η εναλλακτική είναι αληθής και μηδέν διαφορετικά. Αυτό είναι γνωστό ως one-hot encoding
- Αλλά μερικές φορές δεν χρειάζεται να το κάνουμε αυτό επειδή υπάρχει μια φυσική σειρά μεταβλητών, π.χ.
- μικρό=1, μεσαίο=2, μεγάλο=3

Dummy Variable Trap

- Ας υποθέσουμε ότι έχουμε έναν σταθερό όρο και έναν αριθμό από dummy μεταβλητές (ίσες με 0 ή 1).
- Τότε δεν υπάρχει μοναδική λύση γιατί, για οποιοδήποτε C , μπορούμε να προσθέσουμε C στον σταθερό όρο και να αφαιρέσουμε το C από καθεμία από τις dummy μεταβλητές χωρίς να αλλάξουμε την πρόβλεψη.
- Έστω: $Y = a + \dots + b_1 v_1 + b_2 v_2 + \varepsilon$
όπου v_1 and v_2 είναι dummy μεταβλητές έτσι ώστε όταν το ένα είναι 0 το άλλο είναι 1. Αυτό ισοδυναμεί με
$$Y = (a + C) + \dots + (b_1 - C)v_1 + (b_2 - C)v_2 + \varepsilon$$
- Το Regularization (θα συζητηθεί) λύνει αυτό το πρόβλημα.

Overfitting και Variable Selection

- Δηλαδή όσο περισσότερες μεταβλητές τόσο το καλύτερο; ΟΧΙ!
- Οι προσεκτικά επιλεγμένες μεταβλητές μπορούν να βελτιώσουν την ακρίβεια του μοντέλου.
- Αλλά η προσθήκη πάρα πολλών λειτουργιών μπορεί να οδηγήσει σε Overfitting.
- Το πρόβλημα συσχέτισης (Correlation Problem).
- Τα Overfitted μοντέλα περιγράφουν τυχαίο σφάλμα αντί για την υποκείμενη σχέση.
- Τα Overfitted μοντέλα έχουν γενικά κακή προγνωστική απόδοση.
- Η επιλογή μοντέλου μπορεί να αποδειχθεί χρήσιμη για την εύρεση ενός ολιγομεταβλητου (parsimonious) μοντέλου που έχει καλή προγνωστική απόδοση.

Το πρόβλημα συσχέτισης (Correlation Problem)

- Ας υποθέσουμε ότι υπάρχουν δύο χαρακτηριστικά για την εκτίμηση της τιμής του σπιτιού, τα οποία είναι πολύ υψηλά συσχετιζόμενα.

X_1 : Τετραγωνικά μέτρα υπογείου

X_2 : Τετραγωνικά μέτρα πρώτου ορόφου

Η καλύτερη προσαρμογή στο training set μπορεί να είναι:

$$\text{Price} = 10X_1 - 9.5X_2$$

Το παραπάνω κάνει overfit. Ένα καλύτερο μοντέλο θα μπορούσε να είναι το ακόλουθο

$$\text{Price} = 0.5X_1 \quad \text{ή} \quad \text{Price} = 0.5X_2$$

Regularization

- Regularization είναι ένας τρόπος αποφυγής overfitting και, πιθανώς, απλοποίησης του μοντέλου μειώνοντας τον αριθμό των χαρακτηριστικών.
- Εναλλακτικές:
 - Ridge
 - Lasso
 - Elastic net
- **Πρέπει πρώτα να κάνουμε scale τις τιμές των χαρακτηριστικών.**

Ridge regression

- Μειώνει το μέγεθος των συντελεστών παλινδρόμησης επιλέγοντας μια παράμετρο ελαχιστοποιώντας

$$\text{mse} + \lambda \sum_{i=1}^m b_i^2$$

- Κύριος περιορισμός: το traditional inference δεν μπορεί να εφαρμοστεί άμεσα σε εκτιμήσεις ridge regression (να είστε προσεκτικοί) – χρειάζεται «bootstrapping» για την αξιολόγηση της ακρίβειας των εκτιμήσεων ridge regression – (υπολογιστικά χρονοβόρα).
- Γιατί Ridge regression; Ίσως θέλετε να προσαρμόσετε την πολυσυγγραμμικότητα (multicollinearity), αλλά και να διατηρήσετε τους προγνωστικούς παράγοντες/μεταβλητές στο μοντέλο με «σωστούς» συντελεστές (με βάση το πρόσημο).
- L2 regularization
- $\lambda = \text{hyperparameter}$. Μια μεγάλη τιμή στο λ , θα οδηγήσει τα b σε μηδέν

- Υπάρχουν διάφορες επιλογές για επιλογή του λ

A. likelihood function-based criteria (Adj. R^2 , AIC, SBC, ...)

B. cross-validation – χρησιμοποιεί μέρος των training data για να κάνει fit το μοντέλο, και ένα διαφορετικό μέρος για να εκτιμήσει το σφάλμα πρόβλεψης (πχ. mse)

- *k-Fold Cross Validation*: Διαχωρίστε τα δεδομένα σε k ίσου (περίπου) μεγέθους μέρη. Κρατήστε ένα μέρος των δεδομένων για validation και προσαρμόστε το μοντέλο στα υπόλοιπα $k - 1$ μέρη των δεδομένων. Χρησιμοποιήστε αυτό το μοντέλο για να υπολογίσετε το σφάλμα πρόβλεψης για το δεσμευμένο τμήμα των δεδομένων. Κάντε αυτό για όλα τα k μέρη και συνδυάστε τις k εκτιμήσεις του σφάλματος πρόβλεψης και καταλήξτε εκεί όπου ελαχιστοποιείται το cross-validated mean squared error (MSE).

Lasso Regression (μέσω gradient descent)

- Παρόμοια με τη Ridge regression με τη διαφορά μόνο στην ελαχιστοποίηση της

$$\text{mse} + \lambda \sum_{i=1}^m |b_i|$$

- Αυτό έχει ως αποτέλεσμα **την πλήρη εξάλειψη των λιγότερο σημαντικών χαρακτηριστικών**. Άρα κάνει variable selection τροποποιώντας την εκτίμηση του συντελεστή και μειώνοντας ορισμένους συντελεστές στο μηδέν.
- Επιλογές για την εύρεση του λ όπως και με το ridge regression
 - A. likelihood function-based criteria (Adj. R^2 , AIC, SBC, ...)
 - B. cross-validation (π.χ., k-fold cross-validation,...)

Elastic Net Regression (μέσω gradient descent)

- Μεταξύ Ridge και Lasso
- Ελαχιστοποίηση

$$\text{mse} + \lambda_1 \sum_{i=1}^m b_i^2 + \lambda_2 \sum_{i=1}^m |b_i|$$

- Επιλογές για την εύρεση του λ όπως και με ridge και lasso regressions
 - A. likelihood function-based criteria (Adj. R^2 , AIC, SBC, ...)
 - B. cross-validation (π.χ., k-fold cross-validation,...)

Εφαρμογή Iowa House Price

Ανοίξτε το αρχείο `Original_Data.xlsx`. που περιλαμβάνει 80 χαρακτηριστικά για 2908 σπίτια στην Αϊόβα.

- Ο στόχος είναι να προβλέψουμε τις τιμές των σπιτιών στην Αϊόβα με βάση τα χαρακτηριστικά. Παρατηρείτε ότι υπάρχουν `missing values (NA)`.
- Μετά από `data cleaning` καταλήγουμε σε 2908 σπίτια και θα χρησιμοποιήσουμε 47 χαρακτηριστικά.
- Χρησιμοποιούμε 800 παρατηρήσεις στο `training set`, 600 στο `validation set` και 508 στο `test set`.

- Ανοίξτε το αρχείο *Houseprice_data_scaled.xlsx* που περιλαμβάνει τα δεδομένα που θα χρησιμοποιήσουμε στην ανάλυσή μας

❖ 21 numerical variables

❖ 2 categorical variables

Lot area (sq ft)	Number of half bathrooms
Overall quality (scale from 1 to 10)	Number of bedrooms
Overall condition (scale from 1 to 10)	Total rooms above grade
Year built	Number of fireplaces
Year remodeled	Parking spaces in garage
Basement finished sq ft	Garage area sq ft
Basement unfinished sq ft	Wood deck sq ft
Total basement sq ft	Open porch sq ft
1st floor sq ft	Enclosed porch sq ft
2 nd floor sq ft	Neighborhood (25 alternatives)
Living area	Basement quality (6 natural ordering)
Number of full bathrooms	

Αποτελέσματα - Linear Regression

- Ανοίξτε το αρχείο python 5.1 *linear_regression.ipynb*

```
]# MSE at the train set
lr=LinearRegression()
lr.fit(X_train,y_train)
pred=lr.predict(X_train)
print(mse(y_train,pred))
```

0.1140152643124634

```
]# MSE at the validation dataset
lr.fit(X_train,y_train)
pred=lr.predict(X_val)
print(mse(y_val,pred))
```

0.11702499460121657

```
# Create DataFrame with corresponding feature and its respective coefficients
coeffs = pd.DataFrame(
    [
        ['intercept'] + list(X_train.columns),
        list(lr.intercept_) + list(lr.coef_[0])
    ]
).transpose().set_index(0)

lreg_coefficient = pd.DataFrame()
lreg_coefficient["Features"] = X_train.columns
lreg_coefficient['Coef Estimate'] = pd.Series(lr.coef_[0])
print(lreg_coefficient)
```

Ορισμένα weights είναι αρνητικά όταν θα περιμέναμε να είναι θετικά.

Features	Weights	Features	Weights
LotArea	0.079	BrkSide	0.021
OverallQual	0.214	ClearCr	-0.007
OverallCond	0.096	CollgCr	-0.007
YearBuilt	0.161	Crawfor	0.036
YearRemodAdd	0.025	Edwards	-0.001
BsmtFinSF1	0.091	Gilbert	-0.008
BsmtUnfSF	-0.033	IDOTRR	-0.002
TotalBsmtSF	0.138	MeadowV	-0.016
1stFlrSF	0.153	Mitchel	-0.028
2ndFlrSF	0.133	Names	-0.039
GrLivArea	0.161	NoRidge	0.052
FullBath	-0.021	NPkVill	-0.022
HalfBath	0.017	NriddgHt	0.124
BedroomAbvGr	-0.084	NWAmes	-0.052
TotRmsAbvGrd	0.083	OLDTown	-0.026
Fireplaces	0.028	SWISU	-0.004
GarageCars	0.038	Sawyer	-0.018
GarageArea	0.052	SawyerW	-0.028
WoodDeckSF	0.021	Somerst	0.028
OpenPorchSF	0.034	StoneBr	0.063
EnclosedPorch	0.007	Timber	-0.003
Blmngtn	-0.018	Veenker	0.002
Blueste	-0.013	Bsmt Qual	0.011
BrDale	-0.025		

Αποτελέσματα - Ridge Regression

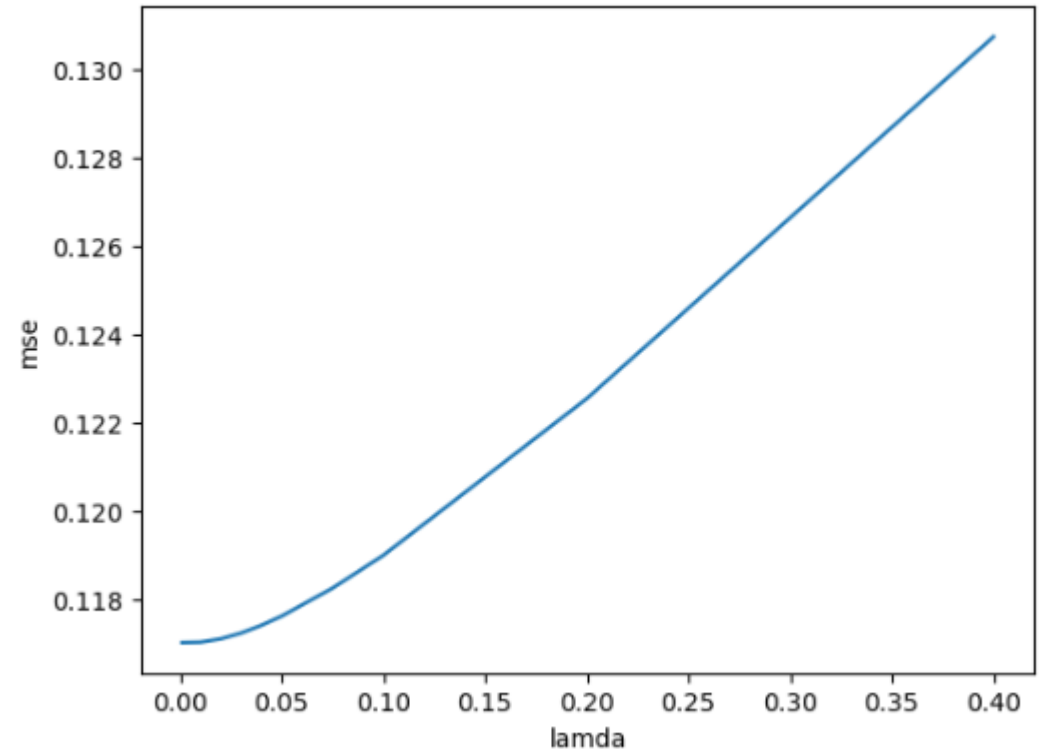
- Ανοίξτε το αρχείο python 5.2 `ridge_regression.ipynb`
- Για το validation set

```
]# Importing Ridge
from sklearn.linear_model import Ridge

# The alpha used by Python's ridge should be the lambda times the number of observations
alphas=[0.001*1800,0.01*1800, 0.02*1800, 0.03*1800, 0.04*1800, 0.05*1800, 0.075*1800,0.1*1800,0.2*1800, 0.4*1800]
#alphas = np.linspace(0, 3.0, num=20)*1800 # from 0 to 3 with 20 numbers inside
#alphas = range(0,2*1800)
mses=[]
for alpha in alphas:
    ridge=Ridge(alpha=alpha)
    ridge.fit(X_train,y_train)
    pred=ridge.predict(X_val)
    mses.append(mse(y_val,pred))
    print(mse(y_val,pred))
```

```
0.11702368844694085
0.11703284346091346
0.11710797319753001
0.11723952924901127
0.11741457158889515
0.11762384068711462
0.11825709631198018
0.11900057469147929
0.12254649996292954
0.1307359968074713
```

```
]plt.xlabel('lamda')
plt.ylabel('mse')
lamdas = [i/1800 for i in alphas]
plt.plot(lamdass, mses)
```



- Όσο μεγαλώνει το λ μεγαλώνει και το mse.
- Για λ από 0 έως 0,1, οι αυξήσεις στο mse είναι μικρές.

- Weights με βάση το Ridge όταν $\lambda=0.1$

Ridge Regression

```
]: # Import Ridge
from sklearn.linear_model import Ridge
# The alpha used by Python's ridge should be the lambda times the number of observ
# Here we produce results for alpha=180 which corresponds to lambda=0.1
ridge = Ridge(alpha=180)
ridge.fit(X_train, y_train)
# DataFrame with corresponding feature and its respective coefficients
coeffs = pd.DataFrame(
    [
        ['intercept'] + list(X_train.columns),
        list(ridge.intercept_) + list(ridge.coef_[0])
    ]
).transpose().set_index(0)

ridgereg_coefficient = pd.DataFrame()
ridgereg_coefficient["Features"] = X_train.columns
ridgereg_coefficient['Coef Estimate'] = pd.Series(ridge.coef_[0])
ridgereg_coefficient
```

Features	Weights	Features	Weights
LotArea	0.071	BrkSide	0.014
OverallQual	0.195	ClearCr	-0.004
OverallCond	0.076	CollgCr	-0.004
YearBuilt	0.104	Crawfor	0.033
YearRemodAdd	0.046	Edwards	-0.004
BsmtFinSF1	0.103	Gilbert	-0.010
BsmtUnfSF	-0.016	IDOTRR	-0.008
TotalBsmtSF	0.111	MeadowV	-0.017
1stFlrSF	0.122	Mitchel	-0.027
2ndFlrSF	0.079	Names	-0.034
GrLivArea	0.159	NoRidge	0.056
FullBath	0.007	NPkVill	-0.023
HalfBath	0.031	NriddgHt	0.119
BedroomAbvGr	-0.060	NWAmes	-0.047
TotRmsAbvGrd	0.081	OLDTown	-0.032
Fireplaces	0.042	SWISU	-0.009
GarageCars	0.047	Sawyer	-0.019
GarageArea	0.060	SawyerW	-0.024
WoodDeckSF	0.025	Somerst	0.027
OpenPorchSF	0.037	StoneBr	0.064
EnclosedPorch	0.003	Timber	0.000
Blmngtn	-0.016	Veenker	0.006
Blueste	-0.012	Bsmt Qual	0.037
BrDale	-0.021		

Διαλέγοντας το λ με βάση k - fold cross-validation (k=10)

Selecting alpha based on k - fold cross-validation

```
##Instead of arbitrarily choosing alpha =4
## , it would be better to use cross-validation to choose the tuning parameter alpha. We can do this us
##Selecting lambda
##Fit Ridge regression through cross validation

from sklearn.linear_model import RidgeCV
regr_cv= RidgeCV(alphas=alphas, fit_intercept=True, cv=10)
model_cv2=regr_cv.fit(X_train,y_train)
model_cv2.alpha_
```

135.0

```
# MSE at the validation dataset
ridge=Ridge(alpha=model_cv2.alpha_)
ridge.fit(X_train,y_train)
pred=ridge.predict(X_val)
print(mse(y_val,pred))
```

0.11825709631198018

```
# MSE at the test dataset
ridge=Ridge(alpha=model_cv2.alpha_)
ridge.fit(X_train,y_train)
pred=ridge.predict(X_test)
print(mse(y_test,pred))
```

0.12017808465302704

```
# Ridge
# import model
from sklearn.linear_model import Ridge
# Here we produce results for alpha=135 which corresponds to lambda=135/1800=0.075
ridge=Ridge(alpha=model_cv2.alpha_)
ridge.fit(X_train, y_train)

# DataFrame with corresponding feature and its respective coefficients
coeffs = pd.DataFrame(
    [
        ['intercept'] + list(X_train.columns),
        list(ridge.intercept_) + list(ridge.coef_[0])
    ]
).transpose().set_index(0)
ridgereg_coefficient = pd.DataFrame()
ridgereg_coefficient["Features"] = X_train.columns
ridgereg_coefficient['Coef Estimate'] = pd.Series(ridge.coef_[0])
ridgereg_coefficient
```

Features	Weights	Features	Weights
LotArea	0.073	BrkSide	0.015
OverallQual	0.199	ClearCr	-0.004
OverallCond	0.079	CollgCr	-0.004
YearBuilt	0.112	Crawfor	0.034
YearRemodAdd	0.042	Edwards	-0.003
BsmtFinSF1	0.103	Gilbert	-0.009
BsmtUnfSF	-0.018	IDOTRR	-0.007
TotalBsmtSF	0.114	MeadowV	-0.017
1stFlrSF	0.125	Mitchel	-0.027
2ndFlrSF	0.084	Names	-0.035
GrLivArea	0.164	NoRidge	0.056
FullBath	0.002	NPkVill	-0.023
HalfBath	0.029	NriddgHt	0.121
BedroomAbvGr	-0.064	NWAmes	-0.048
TotRmsAbvGrd	0.081	OLDTown	-0.032
Fireplaces	0.039	SWISU	-0.009
GarageCars	0.045	Sawyer	-0.019
GarageArea	0.059	SawyerW	-0.025
WoodDeckSF	0.024	Somerst	0.027
OpenPorchSF	0.037	StoneBr	0.064
EnclosedPorch	0.004	Timber	-0.001
Blmngtn	-0.016	Veenker	0.006
Blueste	-0.012	Bsmt Qual	0.033
BrDale	-0.022		

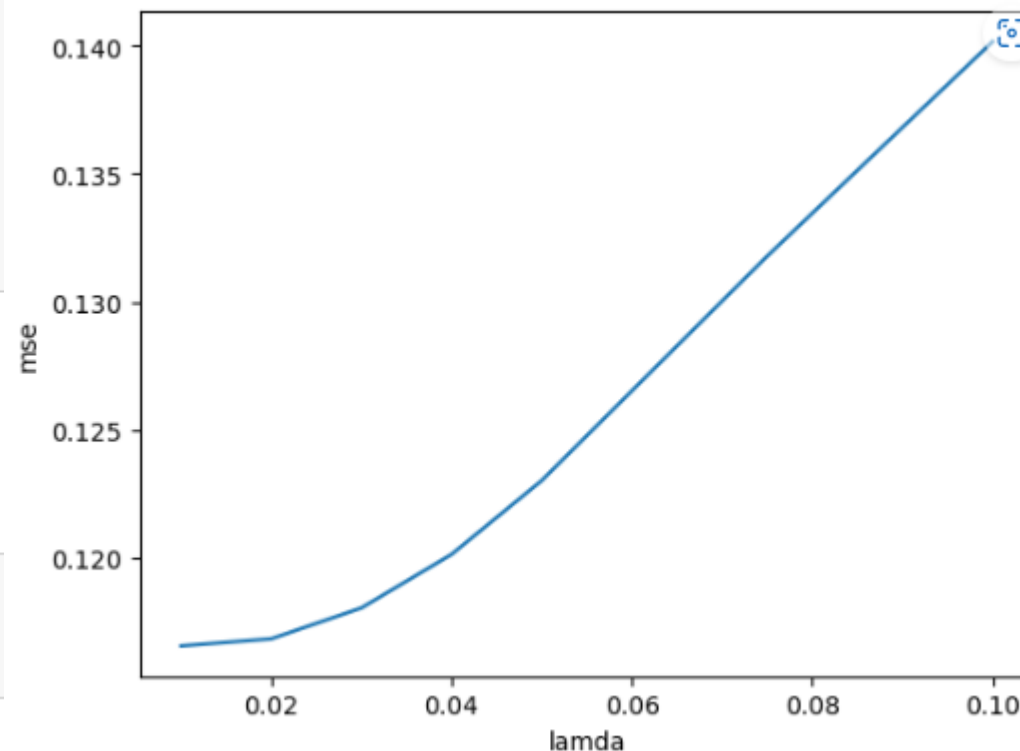
Αποτελέσματα - Lasso Regression

- Ανοίξτε το αρχείο python 5.3 *lasso_regression.ipynb*
- Για το validation set

```
: # We now consider different lambda values. The alphas are half the lambdas
alphas=[0.01/2, 0.02/2, 0.03/2, 0.04/2, 0.05/2, 0.075/2, 0.1/2]
mses=[]
for alpha in alphas:
    lasso=Lasso(alpha=alpha)
    lasso.fit(X_train,y_train)
    pred=lasso.predict(X_val)
    mses.append(mse(y_val,pred))
    print(mse(y_val, pred))
```

```
0.11654751909608799
0.11682687945311097
0.1180334835313203
0.12012836764959005
0.12301536903084047
0.13178576395045638
0.1401719458448378
```

```
: plt.xlabel('lamda')
plt.ylabel('mse')
lambdas = [i *2 for i in alphas]
plt.plot(lambdas, mses)
```



- Non-zero weights με βάση το Lasso **όταν** $\lambda=0.1$ (15 χαρακτηριστικά είναι τα πιο σημαντικά) - Variable selection

```

: # Import Lasso
from sklearn.linear_model import Lasso
# Here we produce results for alpha=0.05 which corresponds to lambda=0.1
lasso = Lasso(alpha=0.05)
lasso.fit(X_train, y_train)
Lasso(alpha=0.05)
# DataFrame with corresponding feature and its respective coefficients
coeffs = pd.DataFrame(
    [
        ['intercept'] + list(X_train.columns),
        list(lasso.intercept_) + list(lasso.coef_)
    ]
).transpose().set_index(0)

lassoreg_coefficient = pd.DataFrame()
lassoreg_coefficient["Features"] = X_train.columns
lassoreg_coefficient['Coef Estimate'] = pd.Series(lasso.coef_)
lassoreg_coefficient

```

Feature	Weight
Lot Area (square feet)	0.04
Overall quality (Scale from 1 to 10)	0.30
Year built	0.05
Year remodeled	0.06
Finished basement (square feet)	0.12
Total basement (square feet)	0.10
First floor (square feet)	0.03
Living area (square feet)	0.30
Number of fireplaces	0.02
Parking spaces in garage	0.03
Garage area (square feet)	0.07
Neighborhoods (3 out of 25 non-zero)	0.01, 0.02, and 0.08
Basement quality	0.02

Διαλέγοντας το λ με βάση k - fold cross-validation (k=10)

```
##We can also select alpha based on the cross-validated ridge regression function, LassoCV().
##Selecting lambda
##Fit Lasso regression through cross validation

from sklearn.linear_model import LassoCV
regr_cv= LassoCV(alphas = None, cv = 10, max_iter = 100000)
model_cv2=regr_cv.fit(X_train,y_train)
model_cv2.alpha_

C:\Users\thano\anaconda3\lib\site-packages\sklearn\linear_model\_coordinate_descent.py:1571: DataConversionWarning: A column
ctor y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)

0.005637136472209141
```

```
# MSE at the validation dataset
lasso=Lasso(alpha=model_cv2.alpha_)
lasso.fit(X_train,y_train)
pred=lasso.predict(X_val)
print(mse(y_val,pred))

0.11649288992474192
```

```
# MSE at the test dataset
lasso=Lasso(alpha=model_cv2.alpha_)
lasso.fit(X_train,y_train)
pred=lasso.predict(X_test)
print(mse(y_test,pred))

0.11886996279710277
```

```
# Here we produce results for alpha=0.006 which corresponds to lambda=0.012
lasso = Lasso(alpha=model_cv2.alpha_)
lasso.fit(X_train, y_train)

# DataFrame with corresponding feature and its respective coefficients
coeffs = pd.DataFrame(
    [
        ['intercept'] + list(X_train.columns),
        list(lasso.intercept_) + list(lasso.coef_)
    ]
).transpose().set_index(0)

lassoreg_coefficient = pd.DataFrame()
lassoreg_coefficient["Features"] = X_train.columns
lassoreg_coefficient['Coef Estimate'] = pd.Series(lasso.coef_)
lassoreg_coefficient
```

Features	Weights	Features	Weights
LotArea	0.077	Blueste	-0.005
OverallQual	0.228	BrDale	-0.014
OverallCond	0.082	BrkSide	0.023
YearBuilt	0.137	Crawfor	0.040
YearRemodAdd	0.032	Edwards	0.005
BsmtFinSF1	0.122	MeadowV	-0.004
TotalBsmtSF	0.107	Mitchel	-0.014
1stFlrSF	0.038	Names	-0.012
GrLivArea	0.302	NoRidge	0.055
HalfBath	0.018	NPkVill	-0.014
BedroomAbvGr	-0.069	NriddgHt	0.128
TotRmsAbvGrd	0.061	NWAmes	-0.035
Fireplaces	0.029	OLDTown	-0.011
GarageCars	0.034	SawyerW	-0.013
GarageArea	0.058	Somerst	0.029
WoodDeckSF	0.019	StoneBr	0.063
OpenPorchSF	0.031	Veenker	0.004
Blmngtn	-0.009	Bsmt Qual	0.015

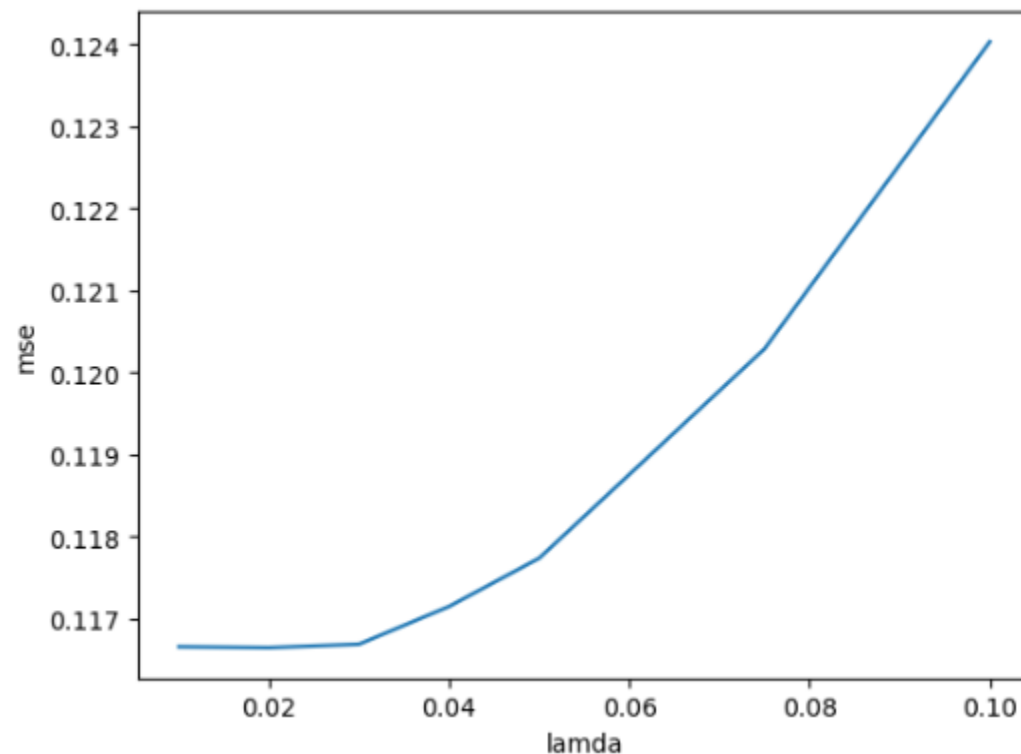
Αποτελέσματα - Elastic Net Regression

- Ανοίξτε το αρχείο python 5.4 `elasticnet_regression.ipynb`
- Για το validation set

```
: #ENET with different levels of alpha and its mse  
# We now consider different lambda values. The alphas are half the lambdas  
alphas=[0.01/2, 0.02/2, 0.03/2, 0.04/2, 0.05/2, 0.075/2, 0.1/2]  
mses=[]  
for alpha in alphas:  
    e_net =ElasticNet(alpha=alpha)  
    e_net.fit(X_train,y_train)  
    pred=e_net.predict(X_val)  
    mses.append(mse(y_val,pred))  
    print(mse(y_val, pred))
```

```
0.11666265584936347  
0.11665113418337304  
0.1166935432601165  
0.1171533394764451  
0.11774118314106564  
0.12028928749167629  
0.12402805248512704
```

```
: plt.xlabel('lamda')  
plt.ylabel('mse')  
lambdas = [i *2 for i in alphas]  
plt.plot(lambdas, mses)
```



- Non-zero weights με βάση το Enet όταν $\lambda=0.1$ (23 χαρακτηριστικά είναι τα πιο σημαντικά) - Variable selection

```
]: # Elastic Net
# import model
from sklearn.linear_model import ElasticNet

# Here we produce results for alpha=0.05 which corresponds to lambda=0.1
e_net = ElasticNet(alpha=0.05)
e_net.fit(X_train, y_train)

# DataFrame with corresponding feature and its respective coefficients
coeffs = pd.DataFrame(
    [
        ['intercept'] + list(X_train.columns),
        list(e_net.intercept_) + list(e_net.coef_)
    ]
).transpose().set_index(0)

enetreg_coefficient = pd.DataFrame()
enetreg_coefficient["Features"] = X_train.columns
enetreg_coefficient['Coef Estimate'] = pd.Series(e_net.coef_)
enetreg_coefficient
```

Features	Weights
LotArea	0.06
OverallQual	0.26
OverallCond	0.03
YearBuilt	0.08
YearRemodAdd	0.06
BsmtFinSF1	0.12
TotalBsmtSF	0.09
1stFlrSF	0.05
GrLivArea	0.29
BedroomAbvGr	-0.01
TotRmsAbvGrd	0.01
Fireplaces	0.03
GarageCars	0.04
GarageArea	0.06
WoodDeckSF	0.01
OpenPorchSF	0.02
Crawfor	0.02
NoRidge	0.04
NriddgHt	0.11
NWAmes	-0.01
Somerst	0.01
StoneBr	0.05
Bsmt Qual	0.03

Διαλέγοντας το λ με βάση k - fold cross-validation

Selecting alpha based on cross-validation

```
##We can also select the alpha using the cross-validated ridge regression function, ElasticNetCV().
##Fit Enet regression through cross validation

from sklearn.linear_model import ElasticNetCV
regr_cv= ElasticNetCV(alphas = None, cv = 10, max_iter = 100000)
#regr_cv=LassoCV(alphas=range(1,50))
model_cv2=regr_cv.fit(X_train,y_train)
model_cv2.alpha_

C:\Users\thano\anaconda3\lib\site-packages\sklearn\linear_model\_coordinate_descent.py:1571: DataConversionWarning: A
ctor y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using.ravel
y = column_or_1d(y, warn=True)

0.011274272944418283
```

```
# MSE at the validation dataset
e_net=ElasticNet(alpha=model_cv2.alpha_)
e_net.fit(X_train,y_train)
pred=e_net.predict(X_val)
print(mse(y_val,pred))

0.11661058532779355
```

```
# MSE at the test dataset
e_net=ElasticNet(alpha=model_cv2.alpha_)
e_net.fit(X_train,y_train)
pred=e_net.predict(X_test)
print(mse(y_test,pred))

0.11919797316720483
```

```
# Elastic Net
# import model
from sklearn.linear_model import ElasticNet

# Here we produce results for alpha=0.011 which corresponds to lambda=0.022
e_net = ElasticNet(alpha=model_cv2.alpha_)
e_net.fit(X_train, y_train)

# DataFrame with corresponding feature and its respective coefficients
coeffs = pd.DataFrame(
    [
        ['intercept'] + list(X_train.columns),
        list(e_net.intercept_) + list(e_net.coef_)
    ]
).transpose().set_index(0)

enetreg_coefficient = pd.DataFrame()
enetreg_coefficient["Features"] = X_train.columns
enetreg_coefficient['Coef Estimate'] = pd.Series(e_net.coef_)
enetreg_coefficient
```

Features	Weights	Features	Weights
LotArea	0.077	Blueste	-0.005
OverallQual	0.227	BrDale	-0.014
OverallCond	0.080	BrkSide	0.023
YearBuilt	0.133	Crawfor	0.040
YearRemodAdd	0.034	Edwards	0.004
BsmtFinSF1	0.121	MeadowV	-0.004
TotalBsmtSF	0.104	Mitchel	-0.014
1stFlrSF	0.043	Names	-0.013
GrLivArea	0.293	NoRidge	0.055
HalfBath	0.020	NPkVill	-0.014
BedroomAbvGr	-0.066	NriddgHt	0.127
TotRmsAbvGrd	0.064	NWAmes	-0.035
Fireplaces	0.030	OLDTown	-0.012
GarageCars	0.035	SawyerW	-0.013
GarageArea	0.058	Somerst	0.029
WoodDeckSF	0.019	StoneBr	0.063
OpenPorchSF	0.031	Veenker	0.004
Blmngtn	-0.009	Bsmt Qual	0.018

Σύνοψη των αποτελεσμάτων

➤ Χωρίς regularization υπάρχει overfitting. Ορισμένα weights είναι αρνητικά όταν θα περιμέναμε να είναι θετικά. Δείτε το παράδειγμα της πολυμεταβλητής γραμμικής παλινδρόμησης.

➤ Με βάση το k-fold cross-validation

- Το Mean squared error για το test set για το Ridge με βάση το 10-fold είναι 12%, άρα εξηγείται το 88% της διακύμανσης.
- Το Mean squared error για το test set για το Lasso και το Enet με βάση το 10-fold είναι 11,9 %, άρα εξηγείται το 88,1% της διακύμανσης.
- Πολύ μικρές διαφορές μεταξύ Ridge, Lasso , ENet !

➤ Υπάρχουν και άλλα κριτήρια που μπορείτε να λάβετε υπόψιν για την επιλογή του λ .