**SNU ECE . Machine Learning**

**Final Term Project**

**AMES Housing Data Supervised Learning**

Kwan Yu Kim, Byeong Hun Kim

**[How To Use]**

1. data folder에“ml\_project\_train.csv” 을 저장한다.

2. 각 regression, classification folder 내에서 “python main.py” 명령어를 실행한다.

3. Additional Test Data의 경우, data folder 에 “ml\_project\_test.csv”의 형태로 저장한다.

(만약 해당 파일이 존재할 경우, 그에 대한 test 결과가 출력됩니다.)

**[Environment Settings]**

Ubuntu 14.04

python 2.7.12

**[Dependencies]**

numpy 1.11.2

scikit-learn 0.18.1

pandas 0.20.2

tensorflow 0.11.0rc0

\* TensorFlow의 경우, 직접 Back Propagation을 구현한 NeuralNetwork와의 결과 비교를 위해서 사용되었습니다.

**[Directory Appendix]**

code/

regression/

-main.py : 실행 파일. Cross Validation 및 Ensemble 알고리즘이 포함되어 있다.

-nn.py : Tensorflow를 통한 Neural Network의 training 및 test.

-customnn.py : 직접 코딩한 Back Propagation을 바탕으로 한 Neural Network.

-pcaknn.py : sklearn 내부의 PCA function을 활용한 k-Nearest Neighbor.

-dataloader.py : training csv를 panda를 통해 읽어들이며, minmaxScaling 및 Normalizing.

**Problem A. [Price Prediction]**

**1. Program your price prediction model that can predict the house price.**

Neural Network에 supervised Learning을 하여 Test Data가 주어졌을 때, 이를 바탕으로 House Sale Price를 예상하는 모델을 만들었다.

1. 먼저, input data에서 nominal 한 값들을 Neural Network에 적합한 1-out 형태로 변형하였다. Oridinal한 값들의 경우, 각각의 대소관계를 정립하여 discrete하게 변형하였다. 이 과정에서 기존 82개 attribute들은 235개로 증가하게 되었다. 최종적으로 전처리 된 데이터들을 minmax scaler를 통해 0~1 사이의 값으로 변환하였다.
2. Network Architecture = [235, 128, 64, 32, 1] 로 설정하였고, 각 layer는 fully connected, activation function은 간단하게 ReLU Function을 채용했다. 각 edge를 대표하게 되는 weight 의 초기값은 Xavier initialization을 활용하였다.
3. Loss Function은 Mean Square Error인 을 활용하였다.
4. Back propagation은 Gradient Descent Algorithm을 응용하여 각각의 weight 를 통해 loss function의 편미분한 값을 learning rate(=0.1) 만큼씩 gradient를 따라 이동하는 형태로 풀이하였다.

**2. Show which variables are comparatively more related to the house price**

PCA 를 통한 dimension reduction을 하였을 때, 각 eigenvector가 eigenvalue만큼의 분산에 영향을 줄 수 있으므로, eigenvalue가 큰 eigenvector에서 coefficient가 큰 attribute들에 주목하였다. 그 결과, "BuiltYear", "SaleCondition", "SaleType", "MSZoning", "Street" 등의 attribute가 dimension reduction에도 상관 없이 분산에 큰 기여를 하게 됨을 알 수 있었다.

**Problem B. [Price Range Classification]**

**1. Program 2 classification models that can classify the house price into 2 classes**

House Price Classification을 위해 두 개의 다른 알고리즘을 사용하는 모델을 사용하였다.

- Neural Network

Problem A에서 활용한 Neural Network를 조금 변형하여 활용하였다. Classification 문제이므로 마지막 output layer로의 function을 softmax cross entropy function을 통해 두 개의 값으로 나누었다. In-sample error rate 는 11%를 얻을 수 있었다.

- PCA + kNN

PCA를 통해서 중요한 dimension만을 남겨둔 채, test Data input과 가장 가까운 k(=5)개의 training Data 의 Class를 확인한 뒤, 그 중 거수로 분류하는 알고리즘이다.

Housing Data의 경우, 부수적인 조건(Roofstyle, Fireplace, KitchenQual)들보다는 중요한 조건들(YearBuilt, LotArea, SaleCondition)에 의해 가격이 좌우될 가능성이 높다. 이러한 특성의 데이터셋에 대해서는 PCA가 큰 효과를 보일 수 있을 것으로 예상하였다.

In-sample error rate는 9.7%로 실제로 위의 Neural Net을 활용한 모델보다 좋은 성능을 보였다.

**2. Show which variables are comparatively more related to the house price class.**

해당 내용은 Problem A.2.와 같은 Training Data를 활용하므로 같은 결과를 얻을 수 있었다.

**3. Improve the model performance of one model.**

Ensemble method를 활용하여, k-fold Cross Validation을 통해 서로 다른 k개의 Training Data Set을 만든 뒤, 이를 학습한 k 개의 Neural Network Classification 모델을 생성하였다. 그 후 Test Data에 대한 결과의 거수를 선택한다.

이러한 방법을 통해서 error rate가 줄어듬은 물론이고, 특정 classification model이 가질 수 있는 편향성을 제거할 수 있었으며, error rate를 매번 비슷하게 유지하는 안정화 효과를 얻을 수 있었다.