ARTIN 2020/09/28 Lab M1 JEMARO name:Yuki Saito student number: 200305D

1: why is it wrong to use all available data for training (no data split) ?

The purpose of machine learning is to estimate good model which fits to unseen data. If we don't split data for training and testing, we can use all the data for training and we cannot prepare the unseen data for the model. If so, we cannot measure how well our model performs on unseen data.

2: What is the effect of increasing the amount of training data ?

Figure1 shows the relationship between training data size and train/test error metrics values. Note that MSE(1st column) is lower is better, and R2 score(2nd column) is higher is better. In the regression model, if we provide more training data, the model performs worse on train set and performs better on test set. It is considered that the increasement of train data cause the higher error, but the model can perform better on unseen data. So, if we increase the amount of training data, the we can enhance the model's generalization capability.

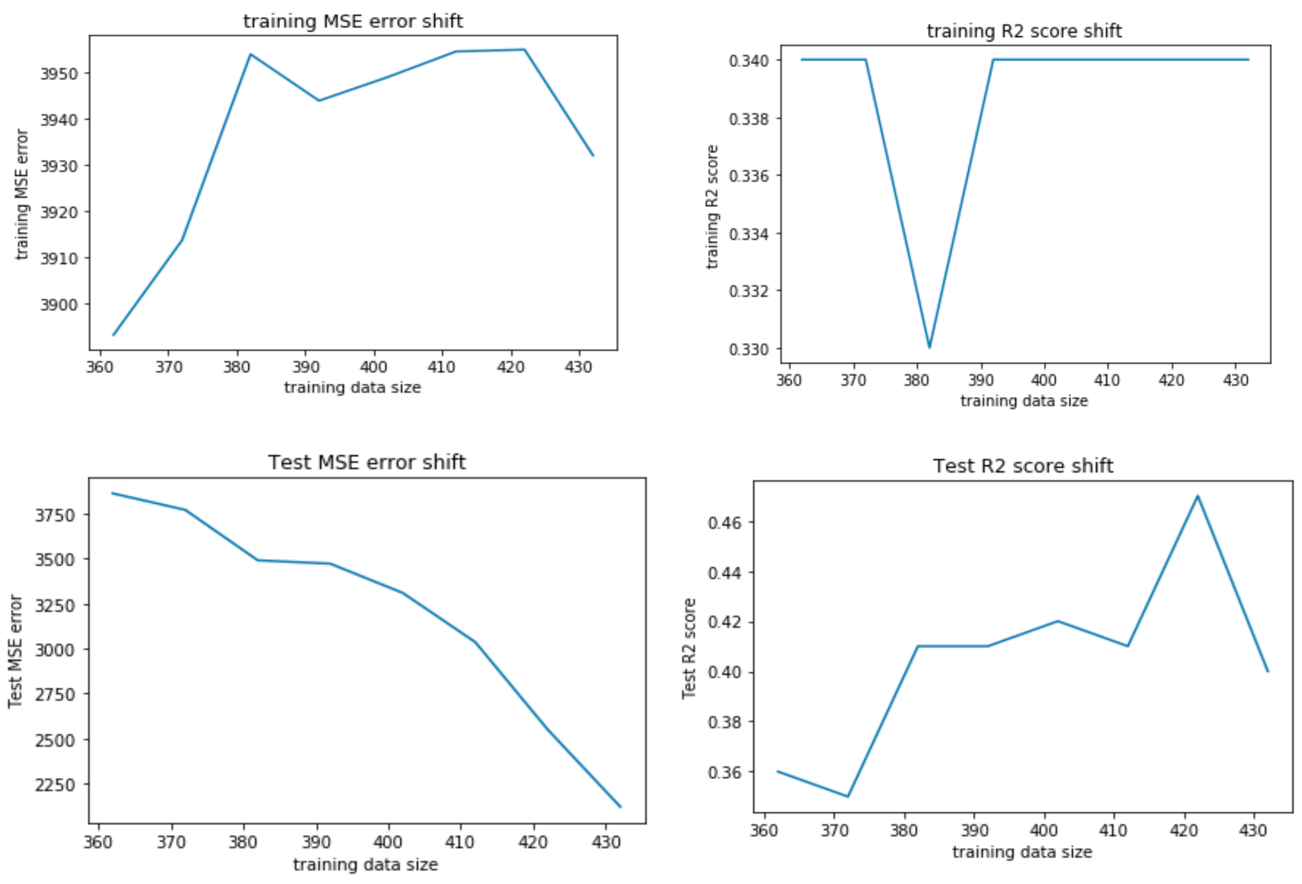

Figure1 relationship between training data size and Train/Test Error metrics

3: Only one feature “bmi” was used during training, is it using more information better?

Figure 2 , Figure 3 shows the Train/Test Error metrics using different features combined with “bmi”. Figure2 shows combination of 2 features with “bmi” and figure 3 shows combination of 3 features with “bmi”. We split the data into train:test = 422:30. The leftmost bar indicates the “bmi” only model. From figure2, 3, we can see it is useful to combine several features for both train, test set. Especially, “bp” and “s5”, are useful rather than other features.

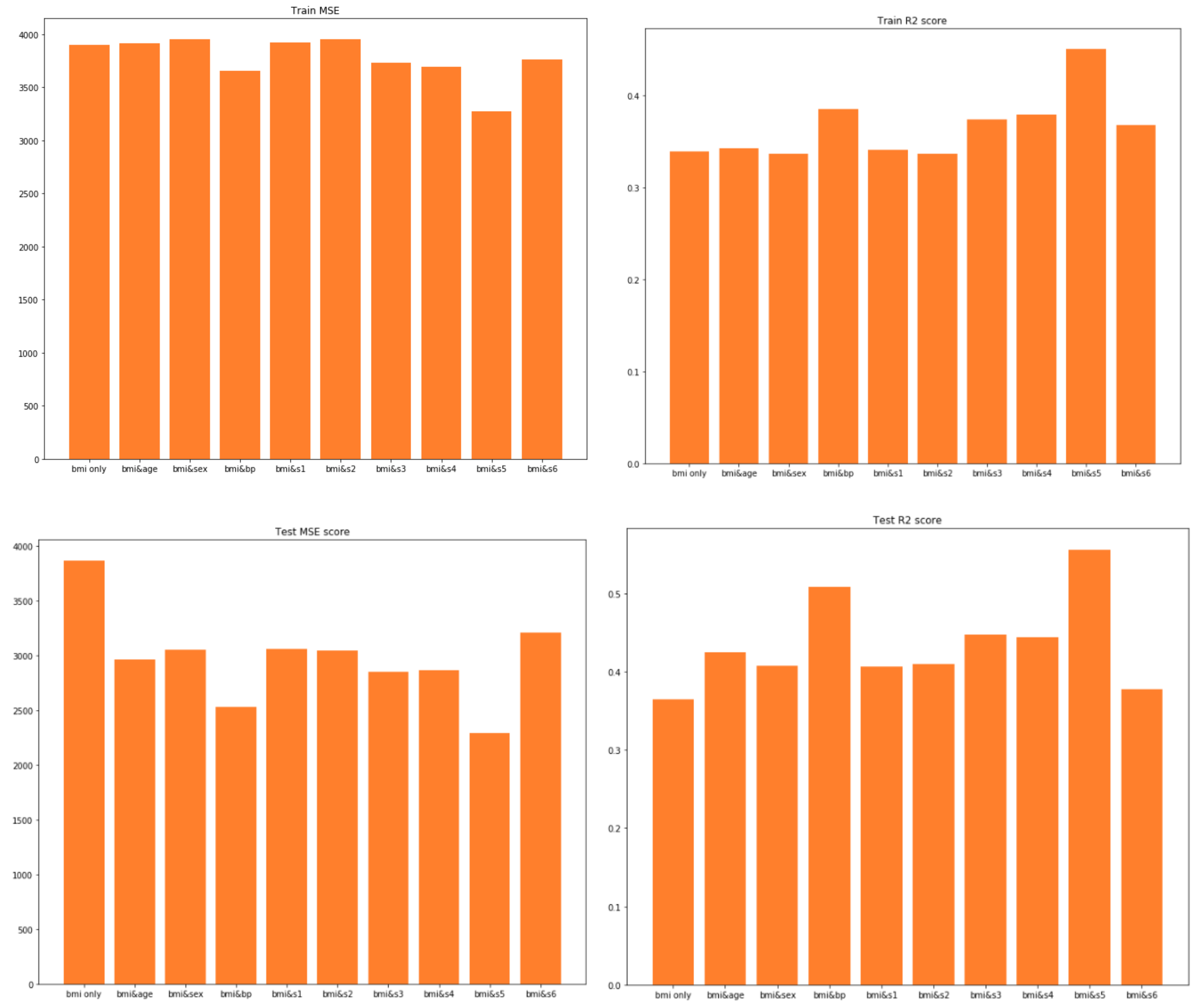

Figure2 Train/Test Error metrics using 2different features(the leftmost bar indicates “bmi” only model)

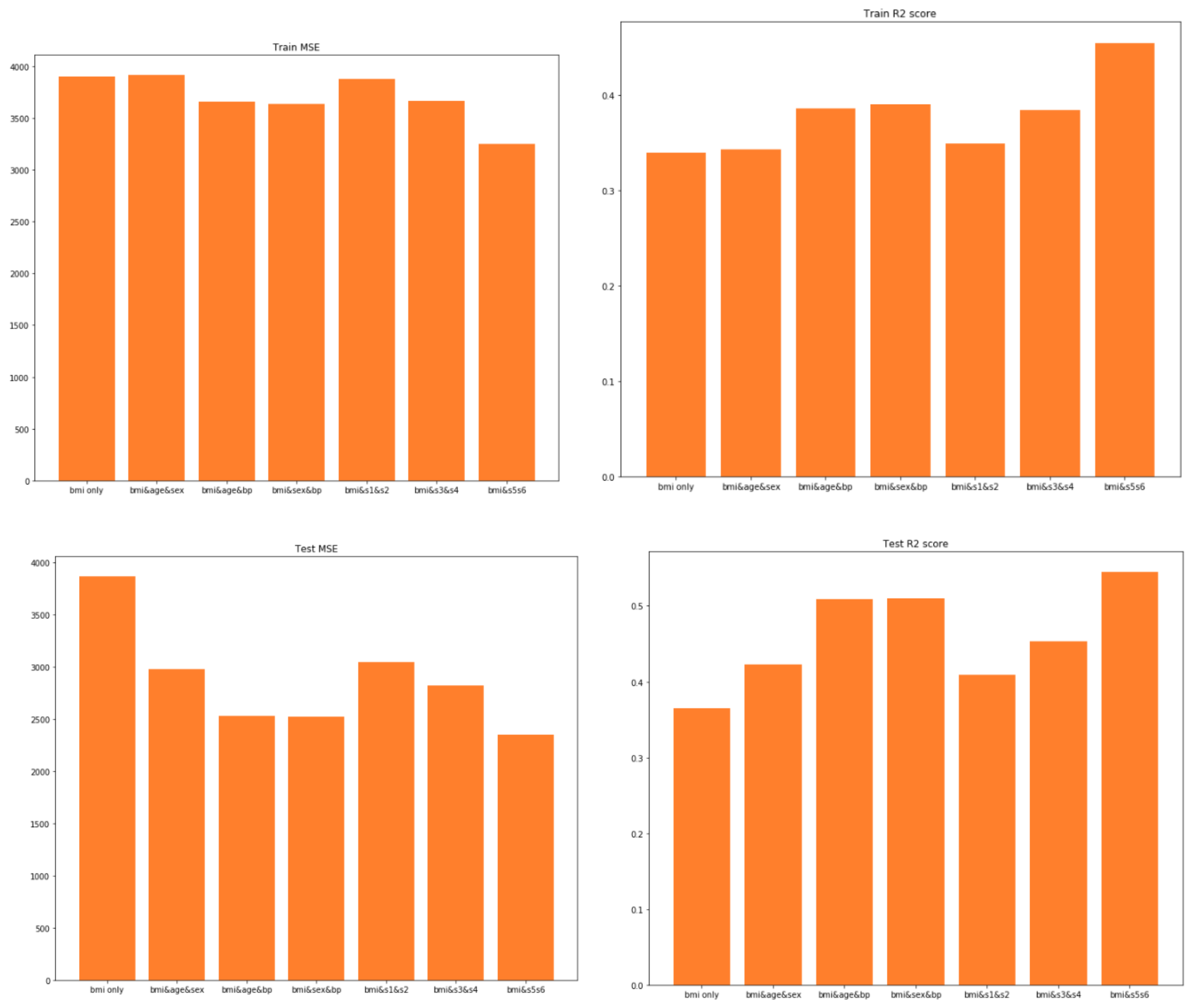

Figure3 Train/Test Error metrics using 3different features (the leftmost bar indicates "bmi" only model)

4; How do we know if the learning was successful in each case ?

In regression model, if we get lower error using MSE, RMSE, AbsRel etc.., we can say the learning was successful. Also, If we get higher score using R2 score, we can say that, too.

5:What model for classification is better , why ?

K-Nearest Neighbors classifier or Naïve Bayes Classifier are better.

Regression task is aimed to approximate a function(line) from input variables to a continuous output variable. So, it helps predict a continuous quantity.

However, if we want to deal with binary classification task like this case, we have to predict discrete labels. So, K-Nearest Neighbors classifier or Naïve Bayes Classifier are desirable.

6: For which of the above models is it useful to split the data in three groups to also consider a validation set?

Validation set is useful for Knn classifier. Validation is used to search the better hyperparameter in machine learning. So, in this case, we have to find the appropriate size for knn.

7; What is the best neighborhood size for knn classifier, what is the appropriate methodology to find good number?

I searched the size of knn using Hold-out validation. I split the dataset into 3 groups using following ratio, train : test: validation = 6: 2: 2. Figure4 shows the relationship between size of knn and the validation accuracy. The horizontal axis indicates the size of k and vertical axis indicates the validation accuracy.
we can see the validation accuracy has maxim values at K=10. So, I considered the k=10 is appropriate size for knn in this case.

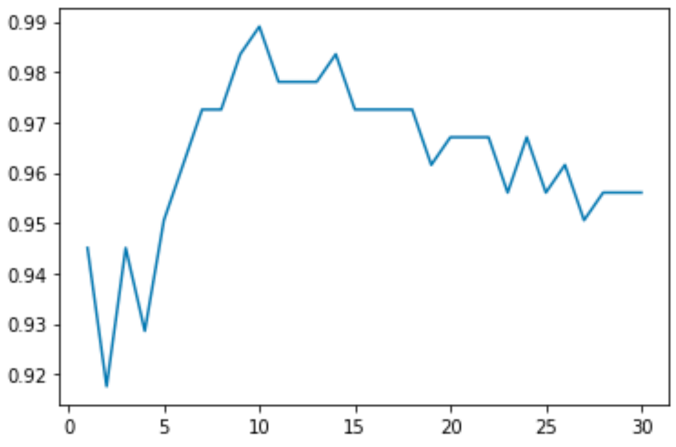

Figure 4　relationship between size of knn and the validation accuracy

And like this, we should use Hold-out validation or Cross validation to approximate appropriate size of k.
[Hold-out validation]
Beforehand, we have to split the dataset into 3 types of data, train data, validation data, test data.
The model is trained by train data and then, check whether the size of k is appropriate or not using validation data. Using this validation, we can search the best size for k.

[Cross validation (K-fold cross validation)]
At first, we split dataset into K groups. For each groups, split data into train and test data. Then fit a model on the train set and evaluate it on the test set. At last, we summarize the evaluation scores of each groups and search the hyperparameter k for knn using this metrics. Compared with Hold-out validation, we can give the model the opportunity to train on multiple train-test splits. So, Cross validation is preferable to Hold-out validation in general.

8: Naive classifiers are probabilistic classifiers. How do we recover the probabilistic information associated to this model? What quantities can we recover?

$$P(y = C_k \mid x_1, x_2 \dots, x_D) = \frac{P(y = C_k) \prod_{d=1}^{D} P(x_d \mid y = C_k)}{\sum_{l=1}^{k} P(y = C_l) P(x_1, x_2 \dots, x_D \mid y = C_l)}$$

At first, we can recover posterior probability for test set $P(y=C_k \mid x_1, x_2 \dots, x_D)$ where k=0, 1

D= 30 using this python code below. (the NumPy array size is 224 × 2)

Priterior_prob = gnbClassifier.predict_proba(X2_test)

Then, we can recover the prior probability for each class $P(y=C_k)$ where k=0, 1 from below.

The values are $P(y=C_0)=0.38709677$, $P(y=C_1)=0.61290323$

Prior_prob_y = gnbClassifier.class_prior_

Also, we get the gaussian distribution parameter for $\prod_{d=1}^{D} P(x_d \mid y = C_k)$. Theta and sigma for gaussian distribution is calculated from below. (the NumPy array size of theta, sigma is 2 × 30)

sigma=gnbClassifier.sigma_

theta=gnbClassifier.theta_

At last, we can recover $\prod_{d=1}^{D} P(x_d \mid y = C_k)$ using the previous gaussian distribution.

```
import math
gaussian_0 = np.sum((1/(sigma[0]*np.power(2*math.pi, 30)))*np.exp(-np.power(X2_test-
theta[0],2)/(2*sigma[0])), 1)
gaussian_1 = np.sum((1/(sigma[1]*np.power(2*math.pi, 30)))*np.exp(-np.power(X2_test-
theta[1],2)/(2*sigma[1])), 1)
gaussian = np.stack([gaussian_0, gaussian_1], 1)
```

9 :we saw in the lecture that linear models were used for regression, while KNN and Naïve Bayes were used for classification. Can we use linear models for classification ? KNN or Naïve Bayes for regression problems?

We can use linear models for classification such as perceptron and SVM etc.
Also, we can use KNN, Naïve Bayes for regression task.