

IE7374 Final Project



Northeastern University
College of Engineering

Machine Learning in Engineering
IE 7374
Summer 2022

Income Classification Project Group 8

Author: Rundong Xu, Yuxi Chen, Haotian Chen
Prof: Ramin Mohammadi,

Income Classification Model

Machine Learning Project

Group: 8

Author Name: Rundong Xu, Haotian Chen, Yuxi Chen

I. Abstract

- Basically, in this project, the objective of this machine learning project is to predict whether a person makes over 50K a year. We are going to focus on building an 'income' classification model. All of these are the prerequisite of default risk modeling.
- For one thing, we apply EDA to understand data from univariate to multivariate analysis, apply Feature Engineering to Bining, encode and scaling, also apply hypothesis testing to do feature selection. For another, we write 7 algorithms scratch without using Sklearn, which includes 1 dimension reduction, and 6 classification algorithms to train our model and evaluate the goal by metrics.

- Finally, we set the 3 dataset scenario including baseline(ad_df) and the other 2 dataset by improvement, and compare the model from metrics perspective, bias-variance-tradeoff perspective, optimization perspective and space and time consumption perspective to conduct model selection and explore the interpretability.

II.Introduction

As far as we are concerned, the census data exerts a significant influence on the development of country. Generally, it is a great measurement and presents a big picture of citizens in the country (eg: its housing conditions and demographic, social and economic characteristics. The information collected includes data on age, gender, country of origin, marital status, housing conditions, marriage, education, employment, etc.)

Typically, all of this information(Data Asset) would be widely used by Bank, Fintech company and IBD for deciding whether to approve their application of card and account, evaluating cardholder's credit, anticipating risk of default and fraud in finance, differentiating the value of bond or interest and commission rate for different applicators and even introducing typical monetization method---Advertisement.

As far as concerned, among six finance metrics including: asset, liabilities, revenue, expense and stock&equity, income is one of most important indicator of individual's living standard and endorsement for one's finance metrics, especially for ability to pay liabilities.

Therefore, exploring the influencing factors of income is of great significance not only for the financial organization, but also for individuals to improve themselves and enterprises, which identifies target consumer groups.

All in all, all of these would study the influence of age, education level, race, occupation and other factors on individual income. Our project takes Adult Data Set of census Data provided by UCI Machine Learning Repository as Data sample.

III.About Data

Our project takes Adult Data Set of census Data provided by UCI Machine Learning Repository as Data sample. Extraction of Adult dataset was done by Barry Becker from the 1994 Census database. A set of reasonably clean records was extracted using the following conditions: ((AGE>16) && (AGI>100) && (AFNLWGT>1)&& (HRSWK>0)).

Besides, it is the multivariable data set. And it has 48842 instances and 14 attributes. Among the 15 indicators, 6 indicators are continuous indicators, and the remaining 9 indicators are discrete indicators.

- 1. Categorical Attributes
 - Workclass: Individual work category
 - Education: Individual's highest education degree
 - Marital-status: Individual marital status
 - Occupation: Individual's occupation
 - Relationship: Individual's relation in a family
 - Race: Race of Individual
 - Sex: Gender
 - Native-country: Individual's native country
- 2. Continuous Attributes
 - Age: Age of an individual
 - education-num: individual's year of receiving education
 - fnlwgt: final weight

The weights on the CPS files are controlled to independent estimates of the civilian noninstitutional population of the US. These are prepared monthly for us by Population Division here at the Census Bureau.
 - Capital-gain
 - Capital-loss
 - Hours-per-week: Individual's working hours per week

III. Data Cleaning:

1. Processing NA

```
1 # We find ? in 'workclass', 'occupation' . 'native.country'
2 ad_df3[ad_df3 == '?'] = np.nan
3 ad_df3.isnull().sum()
```

```

age                0
workclass          2799
fnlwgt             0
education          0
education-num      0
marital-status     0
occupation         2809
relationship       0
race              0
sex               0
capital-gain       0
capital-loss       0
hours-per-week     0
native-country     857
income            0
dtype: int64

```


```

1 # Using Mode to complete Na value
2 for col in ['workclass', 'occupation', 'native-country']:
3     ad_df3[col].fillna(ad_df3[col].mode()[0], inplace = True)

```

IV. Exploratory Descriptive Engineering(EDA)

1. Statistical Analysis

 #For numerical value, we do discriptive analysis.
ad_df3.describe()

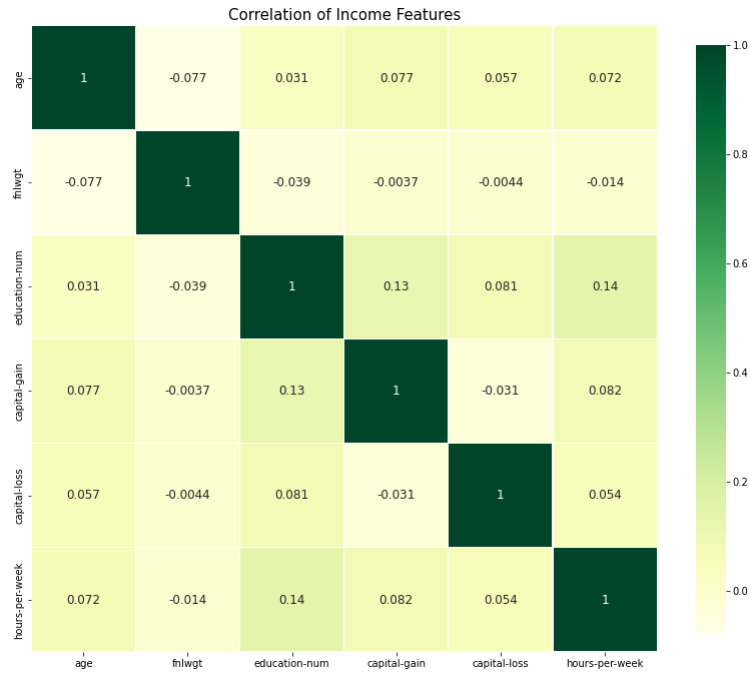
	age	fnlwgt	education-num	capital-gain	capital-loss	hours-per-week
count	48842.000000	4.884200e+04	48842.000000	48842.000000	48842.000000	48842.000000
mean	38.643585	1.896641e+05	10.078089	1079.067626	87.502314	40.422382
std	13.710510	1.056040e+05	2.570973	7452.019058	403.004552	12.391444
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	1.000000
25%	28.000000	1.175505e+05	9.000000	0.000000	0.000000	40.000000
50%	37.000000	1.781445e+05	10.000000	0.000000	0.000000	40.000000
75%	48.000000	2.376420e+05	12.000000	0.000000	0.000000	45.000000
max	90.000000	1.490400e+06	16.000000	99999.000000	4356.000000	99.000000

Takeaway:

- Age: Range from 19 to 90 years, average is 37.
- Education_num: from 1 to 16 ,the avg education level is 10 years.
- hours.per.week:from 1 and 99, and the average is 40 hours.

2. Multivariate Analysis

2.1 Heatmap (Look at the positive and negative correlation)

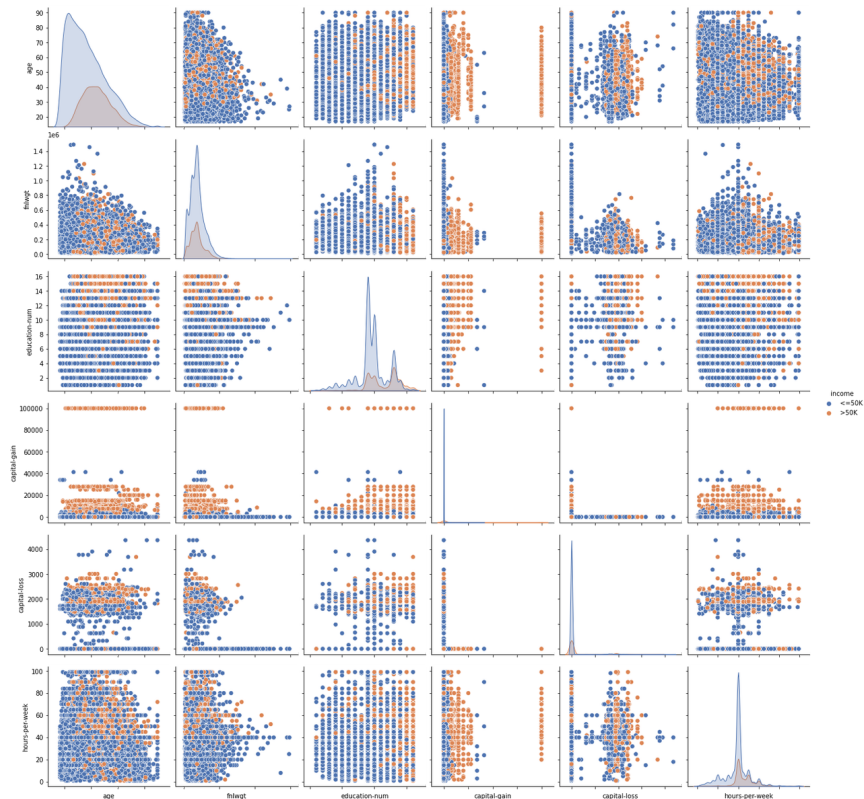


Takeaway:

- 1st: The hours-per-week are highest positive related with capital gain.
- 2nd: The Educational-num are positive related with capital loss
- 3rd: Age vs capital-gain

3. Univariate Analysis and bivariate Analysis:

Pairplot:



- a. **Age:** present right-skew tendency, especially for income<=50k
- b. **Hours-per-week:** it seems to be the present standard tendency.
- c. **Capital-gain and capital-loss:**

- i. there are too many '0' values here, especially for income $\leq 50k$, so both of them are the sparse features. We need to deal with it.
 - ii. There is a lot of polarization, either clustered around 0 or very high-income groups, which also reflects the '2:8' rule of social wealth distribution.
- d. **Education-num**: it seems high education level have more person income $\gg 50K$
- e. **Hours-per-week**: it shows that the more hours per week a person works, the more income they have.



V. Feature Engineering

1. Continuous Feature ---Binning

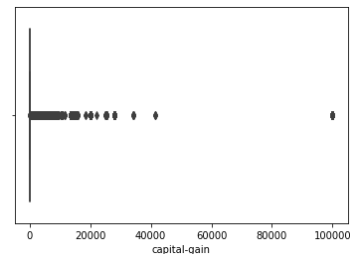
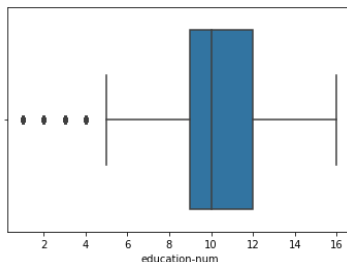
1.1 Feature discreting: Binning

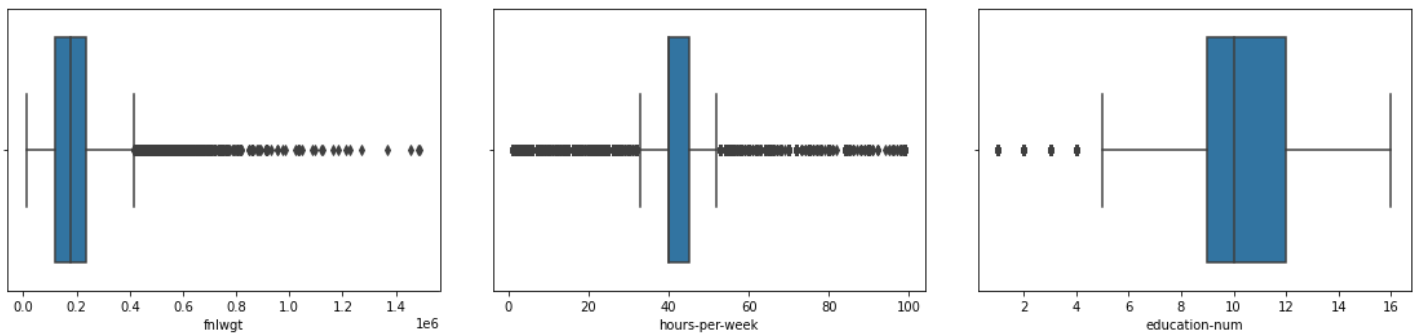
- i. Categorical feature combining: map the every feature have 5 types of catalog
- ii. Equal frequency binning: include as many values in each bin, because we want to keep same data in every binning.

1.2 Deal with Sparse feature

- i. We deal with the problem of too many '0' (sparse feature) by letting 0 is single bin, and others apply equal frequency binning to discreting the feature.

2. Processing Outliers





We set the Low bound and high bound to get rid of outlier and form the `ad_df` dataset. The shape shows that we get rid of 4842 outliers.

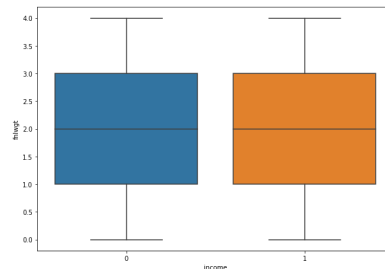
```
# Using quantile to process
LQ = ad_df3["fmlwgt"].quantile(0.01)
HQ = ad_df3["fmlwgt"].quantile(0.99)
ad_df = ad_df3[(ad_df3["fmlwgt"] < HQ) & (ad_df3["fmlwgt"] > LQ)]
```

```
[ ] # Based on the graph and set the Low-bound and High-Bound.
ad_df = ad_df[(ad_df['education-num'] <= 16) & (ad_df['education-num'] >= 4)]
ad_df = ad_df[ad_df['capital-gain'] <= 60000]
ad_df = ad_df[ad_df['capital-loss'] <= 3000]
ad_df = ad_df[(ad_df['hours-per-week'] <= 80) & (ad_df['hours-per-week'] >= 20)]
```

```
ad_df.shape
(44000, 15)
ad_df3.shape
(48842, 15)
```

3. Hypothesis Testing

4.1: `fmlwgt` has too much data, we conduct hypothesis testing.



Null hypothesis: no difference

Alternative hypothesis: exist difference

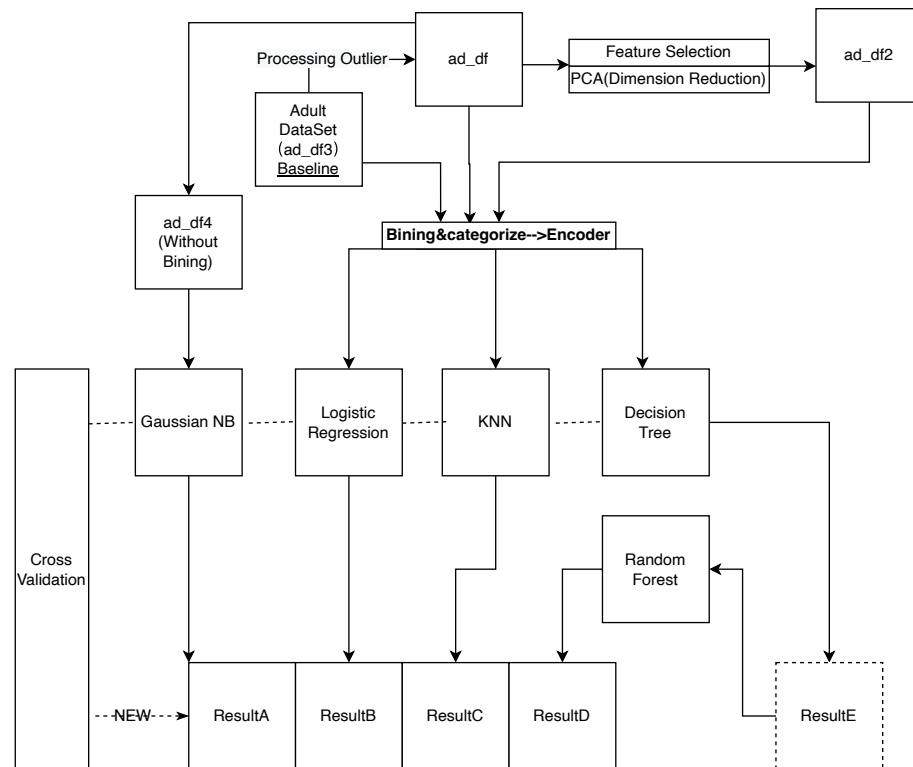
```
ttest 0.8709825672046405
p-value 0.38482368112896215
we accept null hypothesis(no difference between mean of two group of income(>50k and<=50k.)
```

Conclusion: By applying the statistical analysis in two sample t- tests, There no difference between two group of income(>50k and<=50k) which means that the feature 'fmlwgt' has no contribution to the classify income group. Therefore, we decide to drop this feature to form `ad_df2`

4. Label encode

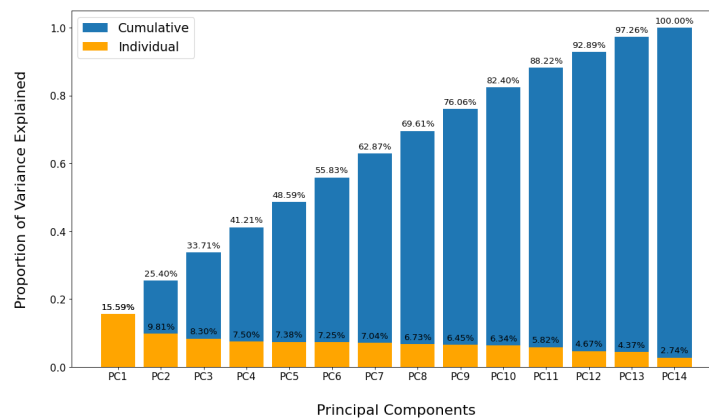
5. Scaling

V. Modeling Method (description of models you have used)



1. PCA(Dimension Reduction)

(PCA) is the process of computing the principal components and using them to perform a change of basis on the data, sometimes using only the first few principal components and ignoring the rest.



Takeaway:

We can see the first 12 variables explained 92.89% variance(Over 90%), Only 7.11% variance would be explained by the last variable.

2. Logistic Regression

2.1 Why do we use it?

- Simple to implement and widely used in industrial problems;
- The amount of calculation is very small, the speed is very fast, and the storage resources are low;

2.2 Design idea

Use the LogisticRegression class that we implemented from HW and class, and the goal of LogisticRegression is to find the weight to then apply it to the gradient descent and find the predicted label

2.3 Problem encountered

- a. cannot apply '(sig - self.y_train).dot(self.X_train)'

2.4 Improvement

- a. change the type of input data into ndarray

3. Gaussian Naive Bayes

3.1 Why do we use it?

- i. The theory is mature and the thinking is simple, which can be used for both classification and regression;
- ii. Can be used for nonlinear classification;
- iii. The training time complexity is $O(n)$;
- iv. No assumptions about the data, high accuracy, not sensitive to outliers

3.2 Design idea

- a. fit distribution for each feature
- b. calculate prior probability for each class

3.3 Problems encountered

- a. There are many features and it is confusing to write code to fit distribution for each feature and each class one by one.
- b. self.fitDistribution can't be used after being saved into a list.

3.4 Improvement

- a. We write a 'for' loop to fit distribution for each feature of two classes.
- b. Store the results in an array.

4. KNN

4.1 Why do we use it?

- a. Simple, easy to understand, high precision; Insensitive to outliers; No data input settings
- b. Downside: High computational complexity and space complexity

4.2 Design idea:

- a. set the k values and calculate the euclidean distances between the sample point and all the train points.
- b. pick top k distances and count the occurrence times of various labels in these K distances
- c. classify the SAMPLE as the label with the most occurrence times.

4.3 Problems encountered

- a. The model used up all the RAM of Colab for the first time, resulting in a forced stop
- b. The second model run took too long

4.4 Improvement

- a. The code itself has many 'For loops' for the second time causing the model to run too long
- b. Through the optimization and rewriting of code, the for loops are reduced without making any errors in the algorithm, thus shortening the time to 3 minutes
 - First model: RAM: out of allowed range Runtime: unable to run since colab crashed
 - Second model: RAM: 1.89gb Runtime: 41min

- Final model: RAM: 2gb Runtime: 3min

```
✓ [76] knn.classify(X2_test, X2_train, y2_train, y2_test)
    knn.score()
```

```
✓ [318] knn1.KNN_classifier(4)
1m      knn2.KNN_classifier(4)
      knn3.KNN_classifier(4)
```



100%		13200/13200 [00:23<00:00, 552.12it/s]
100%		13200/13200 [00:17<00:00, 765.02it/s]
100%		14653/14653 [00:27<00:00, 538.94it/s]

5. Decision Tree

5.1 Why do we use it?

- Simple calculation, easy to understand and strong interpretability
- It is more suitable for processing samples with missing attributes;
- Ability to handle irrelevant features;
- Ability to produce feasible and well-executed results on large data sources in a relatively short period of time.

5.2 Design idea

- use GINI gain to decide where to split the dataset into two parts

At each split decision, we chose that split that has the highest GINI gain. If the GINI gain is non positive, we do not perform the split.

- decide where to split a numeric feature

We first sort all the values to get the means between neighboring values and calculate the GINI gains with each of the means.

- recurve to create the decision tree

5.3 Problems encountered

- It keeps raising error when I increase the max_depth.
- Since there are 13 features of this dataset and each of them has many different values, the process of growing a tree takes a really long time.

5.4 Improvement

- Sometimes we can't get a best feature for splitting before reaching the max depth we want. Therefore, I let it just stop at that depth and predict the nodes at this side as the same class.
- After we binning for continuous features, the time of running code reduces greatly.

6. Random Forest

6.1 Why do we use it?

- Usually, random forest is often the winner of many classification problems (usually a little better than support vector machine), it is fast to train and tunable, and we don't have to worry about adjusting a lot of parameters like a support vector machine, so it has always been more popular than SVM.
- Random forest can greatly reduce overfitting

6.2 Design idea

- a. choose the number of the decision trees
 - i. Suppose that we decide to grow k trees.
 - ii. For each tree, we randomly sample the rows from the original dataset with replacement.
 - iii. And we select a random subset of features to find the best split.
- b. grow the trees
- c. use mode of the predictions of k trees to be the predicted class for test samples

6.3 Problem encountered

- a. When randomly choosing sample points from training data, it always raises error that 'list index out of range'.

6.4 Improvement

- a. After we do feature scaling, the index of X is rearranged, but the index of y is not. We just modify the index of y to make it the same as X's.

VI.Results and Comparison

1. Apply Baseline Comparison

- **ad_df**: Get rid of Outlier
- **ad_df2**: Get rid of Outlier+ drop feature 'fmlwgt' by Hypothesis testing+ drop feature 'native_country'+ 'hours_per_work' by PCA Result.
- **ad_df3**: Do nothing improvement

2. Metrics Perspective:

2.1 Comparison of algorithm within Group

2.1.1 Baseline: ad_df3

	A	B	C	D	E
1	ad_df3	Accuracy	Precision Rate	Recall Rate	F-1 Score
2	Logistic Regression	0.56084	0.208561376	0.30901	0.30901
3	Gaussian NB	0.80515935	0.67925659	0.3281205	0.4424917
4	KNN	0.82004	0.691729	0.426296	0.527504
5	Decision Tree	0.85648	0.8085009	0.5123081	0.5123081
6	Random Forest(Avg-Score)	0.85377	0.7734177	0.5390964	0.6343827

Obeservation:

- Random Forest seems to be the best model among the above 5 algorithms . It has the best F-1 Score. Although its accuracy is 0.85377, the decision tree is 0.85648 .
- If we pursue high accuracy, we could choose the decision tree, but we still need to see bias-variance-trade off in order to evaluate it's generalization ability, given that usually the decision tree is easy to be overfitting.

2.2 Comparing between groups after Improvment

Baseline: ad_df3

	A	B	C	D	E
1	ad_df3	Accuracy	Precision Rate	Recall Rate	F-1 Score
2	Logistic Regression	0.56084	0.208561376	0.30901	0.30901
3	Gaussian NB	0.80515935	0.67925659	0.3281205	0.4424917
4	KNN	0.82004	0.691729	0.426296	0.527504
5	Decision Tree	0.8564799	0.8085009	0.5123081	0.5123081
6	Random Forest(Avg-Score)	0.85377	0.7734177	0.5390964	0.6343827

2.2.1 After Processing Outliers

Obeservision:

- ### 2.2.2 After Feature Selection(Hypothesis testing+PCA)

Obeservision:

- ### 2.3 Conclusion(Algorithm Perspective):

- i. Accuracy between algorithm

- The Decision tree, Random forest and KNN usually occupy the Top 3.
- The lowest accuracy is Logistic Regression, because LR have these cons:
 - when the feature space is large, the performance of logistic regression is not very good;
 - It is easy to under-fit, and the general accuracy is not very high
 - Does not handle a large number of multi-class features or variables well;
- **We could choose Decision tree, which has relatively good metrics, in general. But we still need to look at it's generalization ability, because it easily causes overfitting.**

ii. Other Metrics Comparison

- **Precision rate**
 - When the cost of False Positive (FP) is very high (the consequences are serious), that is, when it is expected to avoid generating FP as much as possible, it should focus on improving the Precision index. We should choose **decision tree 0.8085009 precision rate**

- **Recall:**

- When the cost of False Negative (FN) is very high (the consequences are serious), and you want to avoid generating FN as much as possible, you should focus on improving the Recall indicator. We should choose logistic regression **0.84 recall rate, or decision tree 0.766324 recall rate.**

- **F1-Score:**

- If we want to keep Precision rate and Recall tradeoff, usually, F1- Score are important in common knowledge. So we should choose **logistic regression 0.84032 F1-Score, or choose random forest 0.6343827 F1-Score.**

3. Bias-variance-trade-off:

$$E(f; D) = \underbrace{bias^2(x)}_{\text{red box}} + \underbrace{var(x)}_{\text{blue box}} + \underbrace{\varepsilon^2}_{\text{green line}}$$

$bias^2(x) = (\bar{f}(x) - y)^2$

$var(x) = \mathbb{E}_D \left[(f(x; D) - \bar{f}(x))^2 \right]$

Error= inaccurate estimation due to too simple model (Bias) + larger variation space and uncertainty due to too complex model (Variance) + noise

a. Fitting situation

- i. Low bias, high variance---- overfitting
- ii. High bias, low variance ----underfitting
- iii. High bias, high variance ----- underfitting
- iv. Low bias, low variance ---- good fitting

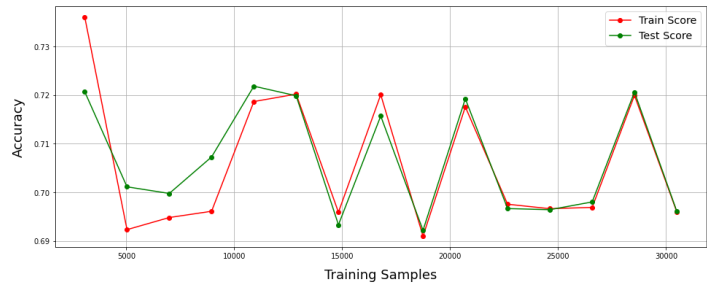
b. For accuracy and error:

- i. high bias: train accuracy is low
- ii. high variance: train accuracy is high, test accuracy is low.

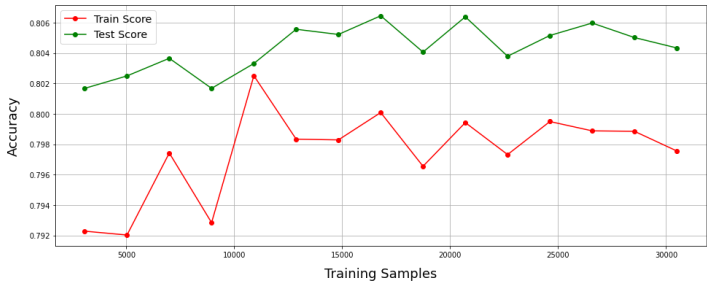
3.1 Learning Curve

IE7374 Final Project - Lark云文档

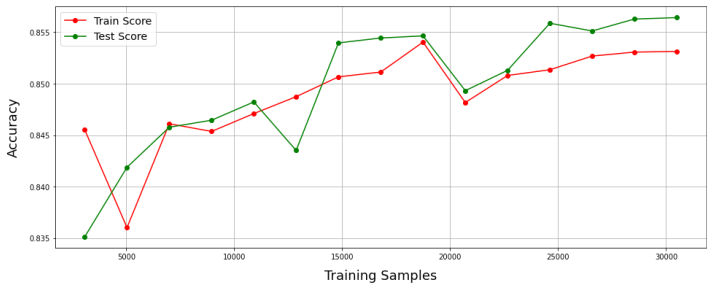
Logistic Regression



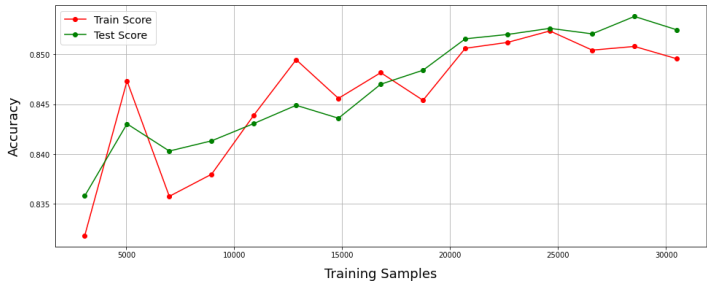
Gaussian Naive Bayes



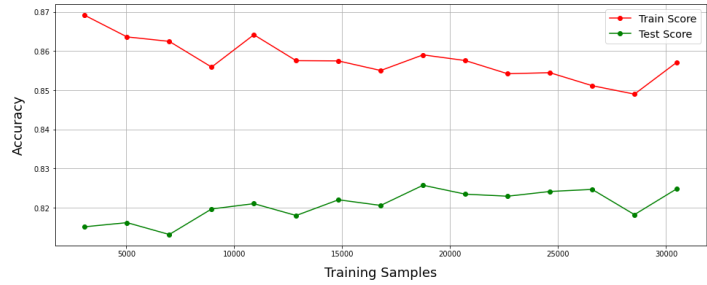
Decision Tree



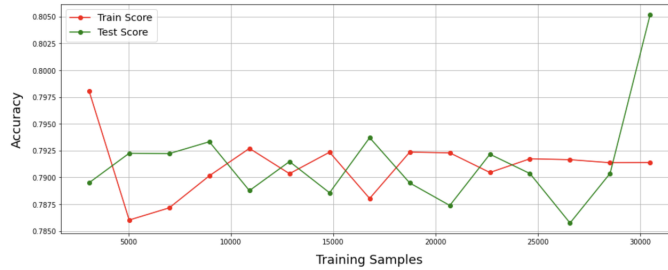
Random Forest



KNN



Gaussian Naive Bayes



3.1.1 Takeaway:

- Based on plot, we can see the best fitting model is Random Forest and Decision tree.
- We can see Naive bayes test rate is high. It may be because of sampling bias, such as the samples in the test set are easy to predict. It is coincidence.

3.1.2 Generalization Ability Thinking:

Usually the model is trained by minimizing the training error, but the real concern is the test error. Therefore, the generalization ability of the model is evaluated by test error.

- The training error is the average loss of the model over the training set, and its size is meaningful, but not intrinsically important.
- The test error is the average loss of the model on the test set, which reflects the predictive ability of the model on the unknown test data set. The high 'mean value' means it changes the least with test size

Therefore, based on the curve, we could conclude that Random Forest has the best generalization ability than others.

3.2 Conclusion2(bias-variance-tradeoff Perspective):

- Naive Bayes: high bias, low variance.
- KNN high variance, and it seems overfitting, but its learning curve not sharp,
- Overfitting is usually caused by three reasons: 1. The model is too complex and there are too many parameters, 2. The generalization ability of the model is not enough, 3. The data is too small
- For example: KNN>NB in bias-variance-tradeoff**
 - In small dataset, high bias/low variance classifier (e.g., Naive Bayes NB)> Low bias/high variance classifier (KNN), cause it would cause overfitting.
 - With Training data improved, ability of prediction would be improved, then bias would be lower, so Low bias/high variance classifier (KNN) (It has low asymptotic error)>NB

4. K-fold Cross Validation:

Usage:

For example, a random variable X , the expected value is $E(X)$, CV does not change this expectation, but CV would only lower the $Var(X)$

4.1 Score Before 5 fold cross Validation

		Logistic Regression	Gaussian Naive Bayes	Decision Tree	Random Forest	KNN
ad_df	Accuracy	0.688561	0.789621	0.835682	0.836242	0.819773
	Precision	0.439449	0.635573	0.755799	0.739229	0.724590
	Recall	0.840324	0.393948	0.517376	0.553775	0.463451
	F1 Score	0.577101	0.486407	0.614263	0.629451	0.565321
ad_df2	Accuracy	0.696288	0.789167	0.833636	0.835394	0.823485
	Precision	0.445756	0.666068	0.766325	0.740039	0.716309
	Recall	0.825944	0.333433	0.492211	0.544787	0.500000
	F1 Score	0.579019	0.444400	0.599416	0.624704	0.588920
ad_df3	Accuracy	0.560841	0.805159	0.856480	0.853614	0.820037
	Precision	0.208561	0.679257	0.808501	0.787708	0.691729
	Recall	0.309007	0.328120	0.512308	0.518853	0.426296
	F1 Score	0.249037	0.442492	0.627194	0.625431	0.527504

Best Accuracy

	Best Acc	Dataset
Logistic Regression	0.696288	ad_df2
Gaussian Naive Bayes	0.805159	ad_df3
Decision Tree	0.856480	ad_df3
Random Forest	0.853614	ad_df3
KNN	0.823485	ad_df2

4.2 After CV

	Accuracy	Precision	Recall	F1 Score
Logistic Regression	0.709273	0.598386	0.512280	0.417035
Gaussian Naive Bayes	0.798526	0.662899	0.321926	0.433363
Decision Tree	0.851188	0.795933	0.509542	0.620708
Random Forest	0.851286	0.783692	0.523697	0.627317
KNN	0.830159	0.729267	0.499862	0.592932

	K-fold Acc	Best Acc	Dataset
Logistic Regression	0.709273	0.696288	ad_df2
Gaussian Naive Bayes	0.798526	0.805159	ad_df3
Decision Tree	0.851188	0.856480	ad_df3
Random Forest	0.851286	0.853614	ad_df3
KNN	0.830159	0.823485	ad_df2

4.3 Takeaway:

- We can see that accuracy is increased after CV in Logistic Regression.
- Others would be decreased a little, maybe because we use 5 fold, so the sample size is decreased.

5. Cosumption Perspective:

5.1 Time Consumption

	ad_df1	ad_df2	ad_df3
	run time	runtime	runtime

Logistic Regression	7 sec	7 sec	8 sec
Gaussian Naive Bayes	46 sec	38 sec	50 sec
Decision Tree	7 sec	5 sec	14 sec
Random Forest	53sec	47 sec	56 sec
KNN	1h56min - > 23 sec	17 sec	27 sec
K-Fold Cross Validation	5 min	4 min	7 min

5.2 RAM Space

	All ad_df RAM
Logistic Regression	2 gb
Gaussian Naive Bayes	3 gb
Decision Tree	2 gb
Random Forest	2.5gb
KNN	2gb

a. Takeaway

i. KNN:

It has large amount of calculation, and face sample imbalance problem (i.e. some classes have a large number of samples while others have a small number). It requires a large amount of memory(RAM).

VII.Discussion