รายงานวิศวกรรมศาสตร์
มหาวิทยาลัยเชียงใหม่
Engineering Report
Chiang Mai University

Report

Kaggle Classification Project

Komcharn Nitrat

Faculty of Engineering, Department of Computer engineering, Chiang Mai, 50200

Email: komcharn_n@cmu.ac.th, Student ID: 630615018

Abstract

In this project, I developed the machine learning models to predict whether the customer will applied for the credit card or not base from those information in the dataset that provide by from Kaggle.com. I have evaluated each model that have the best result for training and testing by divide into 2 sections, the first section is models without hyperparameter tuning and the second section is models with hypermeter tuning. The criterion of evaluation is based on those performance score by did the priority variables candidate sort from recall(FN) which is our potentially customer that wish to apply for the card, but the bank hasn't promoted to them then follow by accuracy, precision, F1, and AUC score.

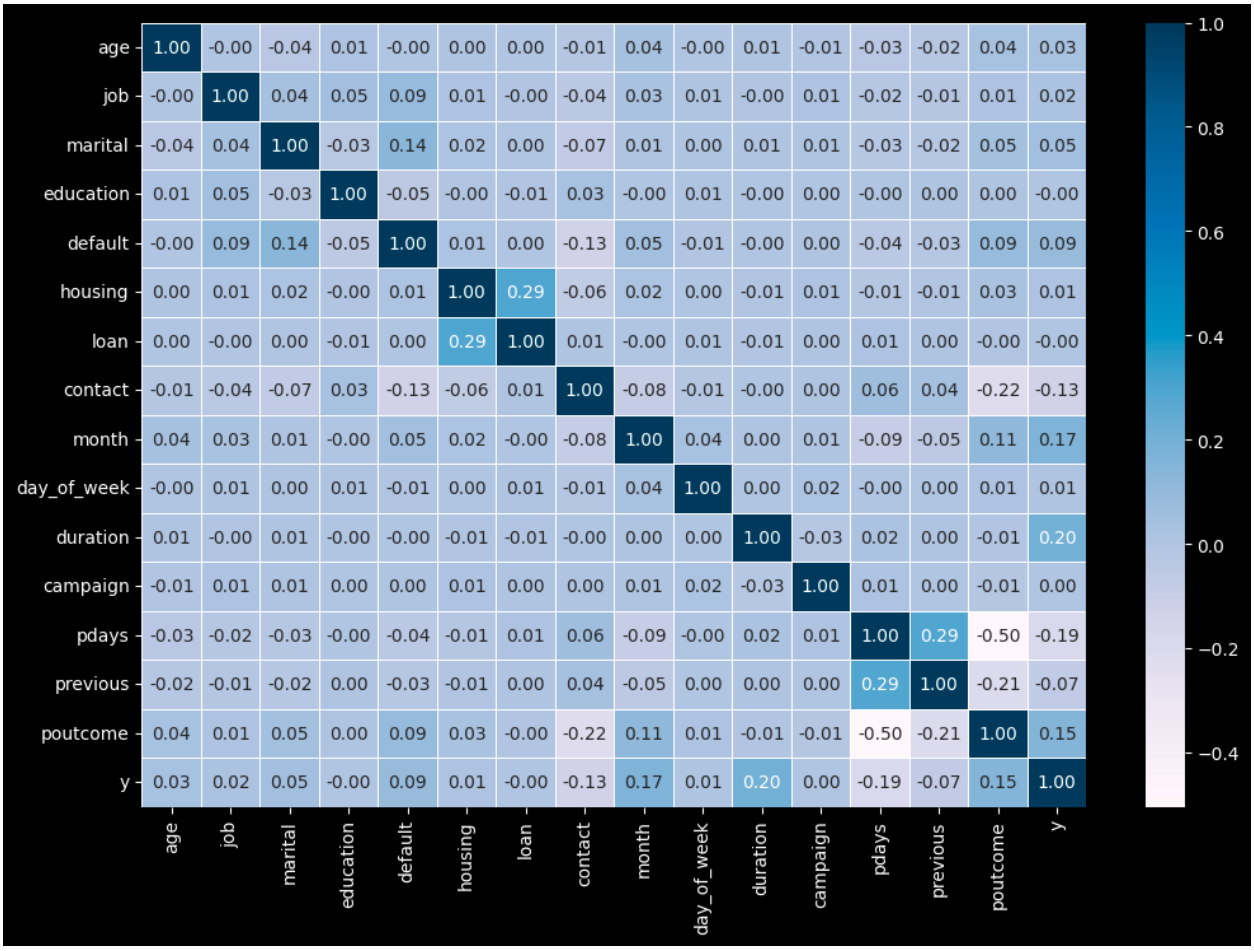
Table of content

Classification Problem

The problem title is Chance of customer to applied credit card base from dataset that contain some useful information like age, job, education, housing, loan, pday, y(as yes and no) and so on by analyze those features and mapping correlation then transform those object type features into numerical variable and concatenate them together to do modeling and find the best model for this problem dataset. For an evaluation to choose the best model I have done by priority candidate base on recall(FN) first because they were potential customer that if we would promote the card to them they were apply so the main goal is the minimize FN by increasing recall, then follow by other variables.

The reason that I choose this classification problem is there are vary features to try and play with, need to do transforming of object type features into numerical variable, and have lot of information in the dataset.

Correlation Heatmap



General variable setting

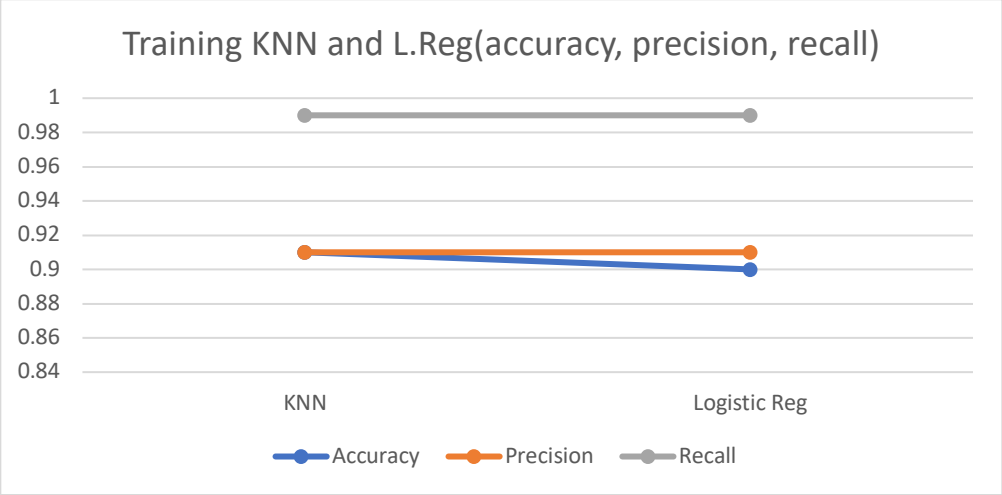
- **Train-Test split:** 80:20
- **Random state:** 42

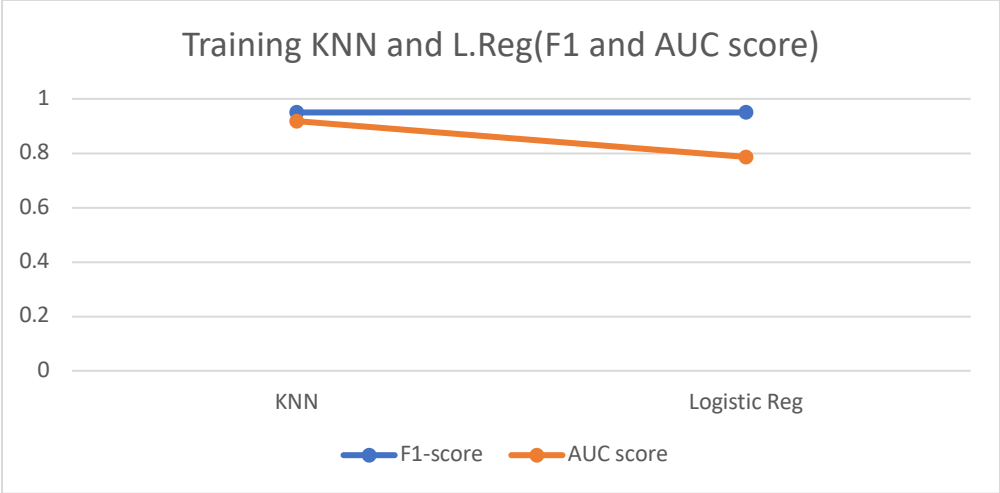
* Note that all models are used pipeline.

Train and Test result without hyperparameter tuning

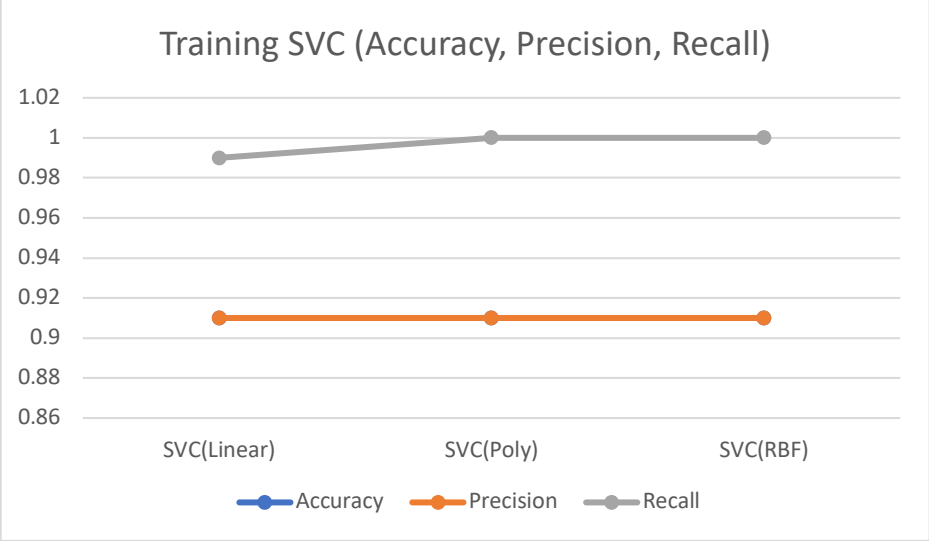
Training

Model	Accuracy	Precision	Recall	F1-score	AUC score
KNN	0.91	0.91	0.99	0.95	0.9181
Logistic Reg	0.90	0.91	0.99	0.95	0.7864
SVC(Linear)	0.91	0.91	0.99	0.95	0.5315
SVC(Poly)	0.91	0.91	1.00	0.95	0.5632
SVC(RBF)	0.91	0.91	1.00	0.95	0.5390

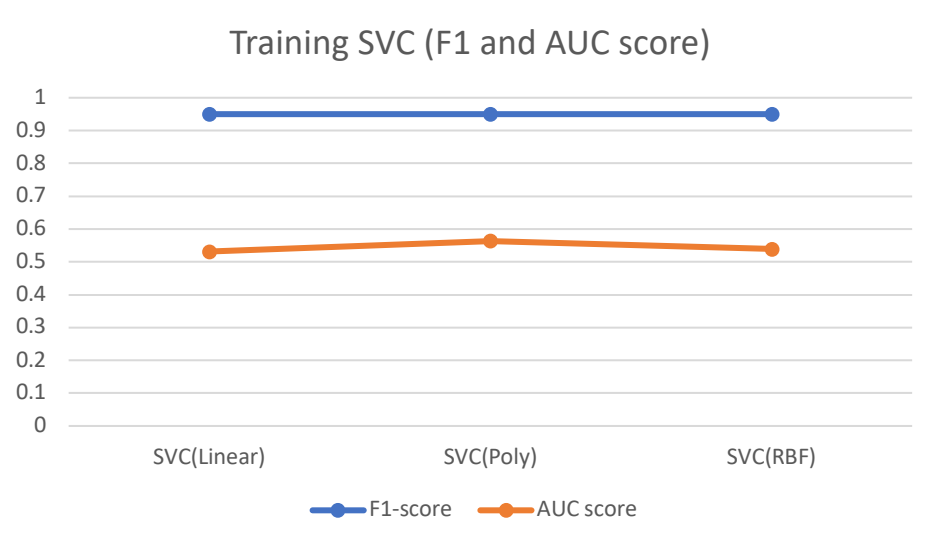




Candidate: KNN, since recall value is the same, but KNN has much more higher AUC score.

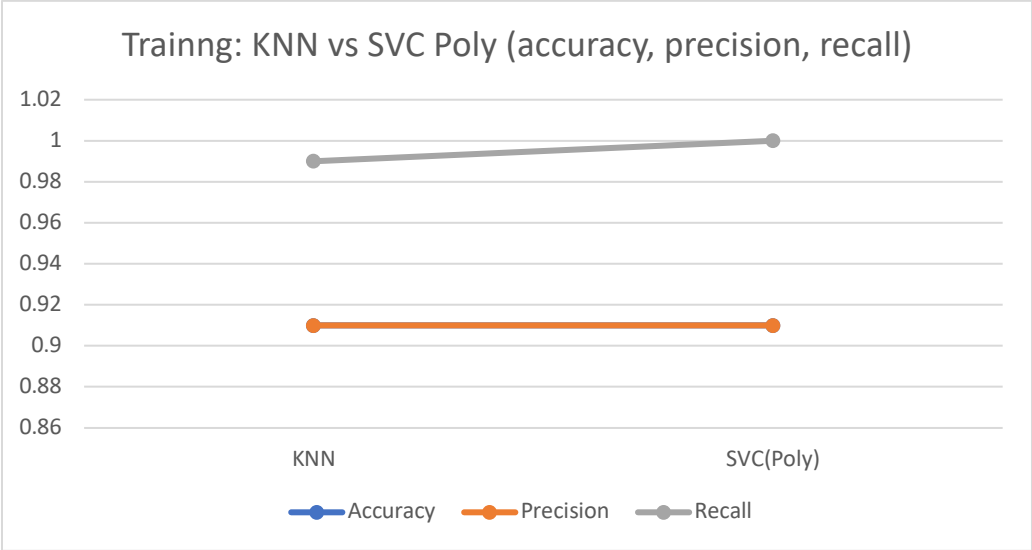


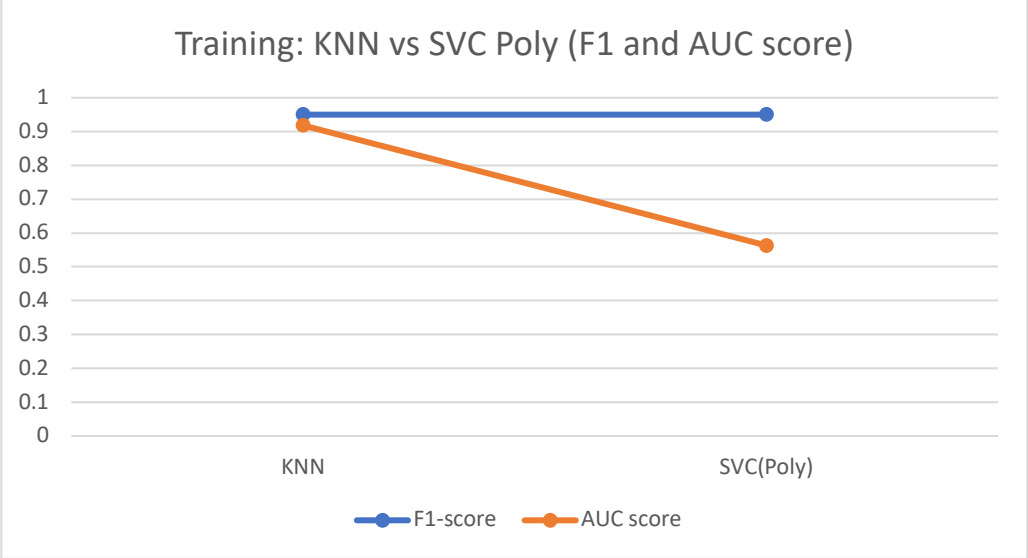
* Note that all SVC models got the same **accuracy** and **precision** score



Candidate: SVC(Poly), both 3 SVC models got same precision, but SVC poly and RBF did better job and since F1-score is also the same, but SVC poly got higher AUC score.

Compare Training result between KNN and SVC(Polynomial)

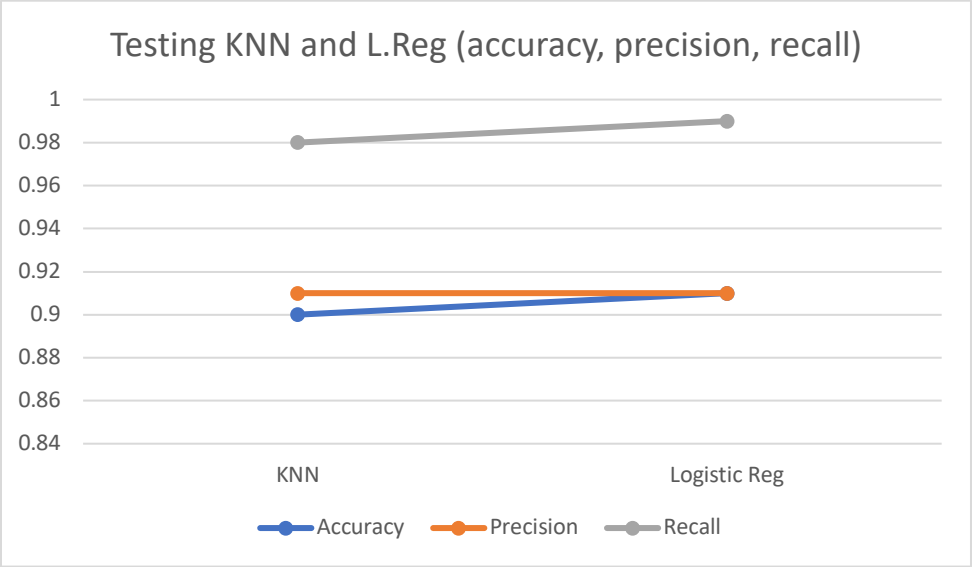


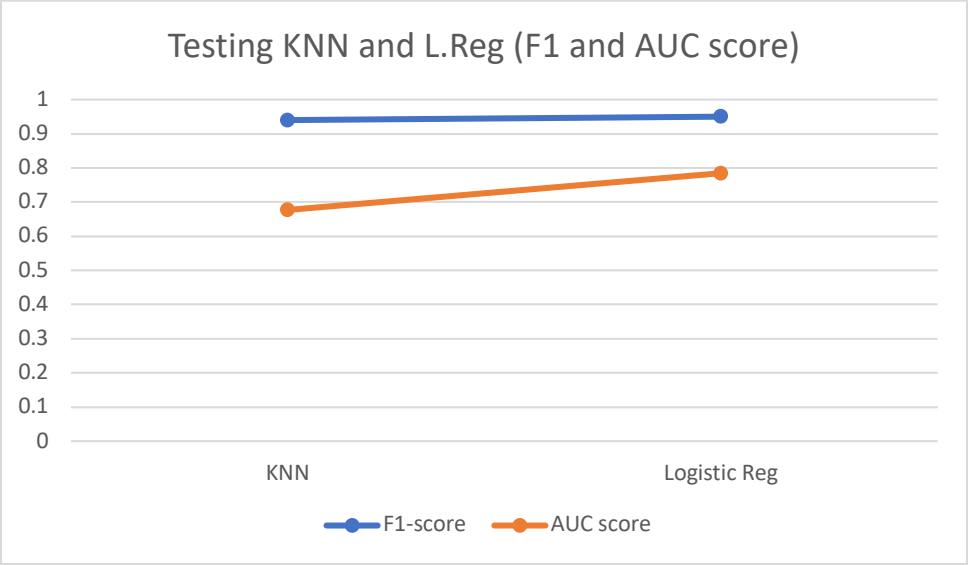


Best model for training: KNN, both KNN and SVC(Poly) has equal accuracy and precision, even though SVC(Poly) got higher recall score but for AUC score SVC(Poly) is not satisfy then the best model for training is KNN.

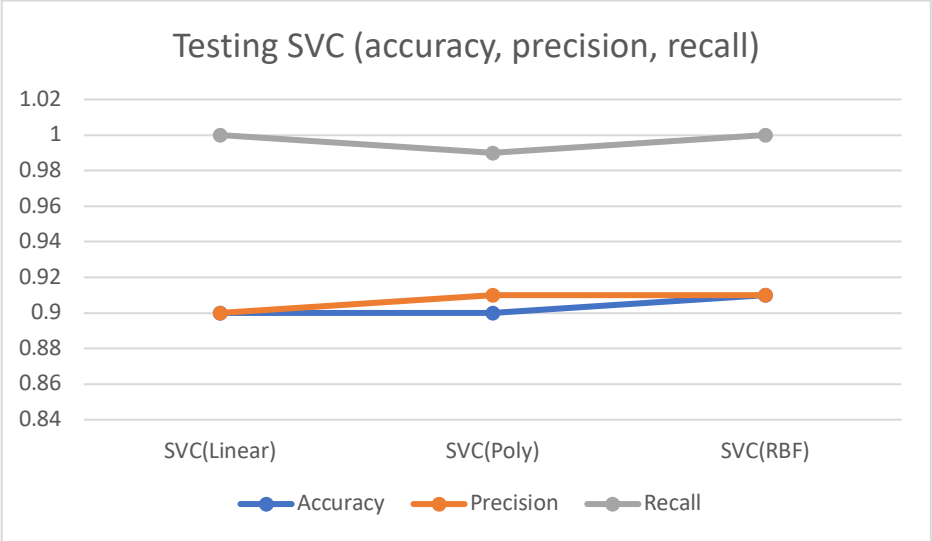
Testing

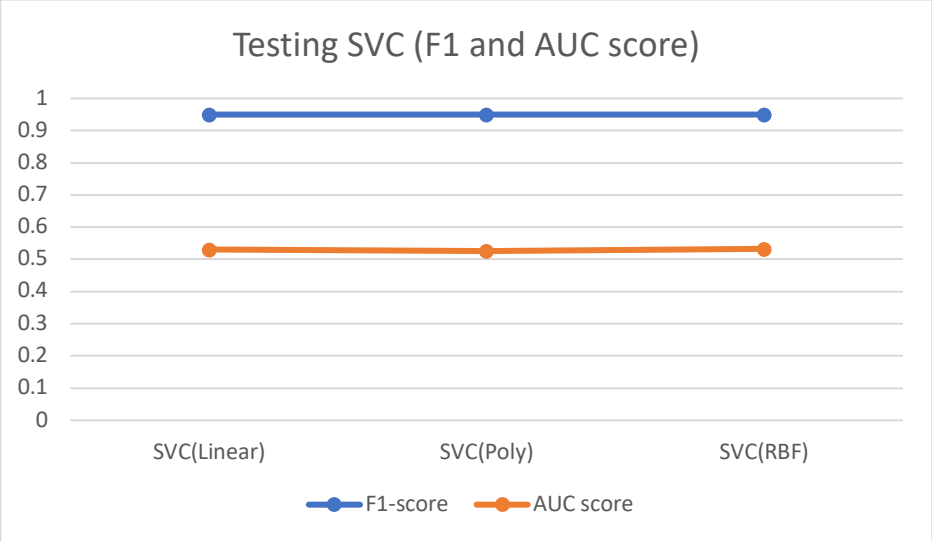
Model	Accuracy	Precision	Recall	F1-score	AUC score
KNN	0.90	0.91	0.98	0.94	0.6769
Logistic Reg	0.91	0.91	0.99	0.95	0.7845
SVC(Linear)	0.90	0.90	1.00	0.95	0.5298
SVC(Poly)	0.90	0.91	0.99	0.95	0.5258
SVC(RBF)	0.91	0.91	1.00	0.95	0.5327





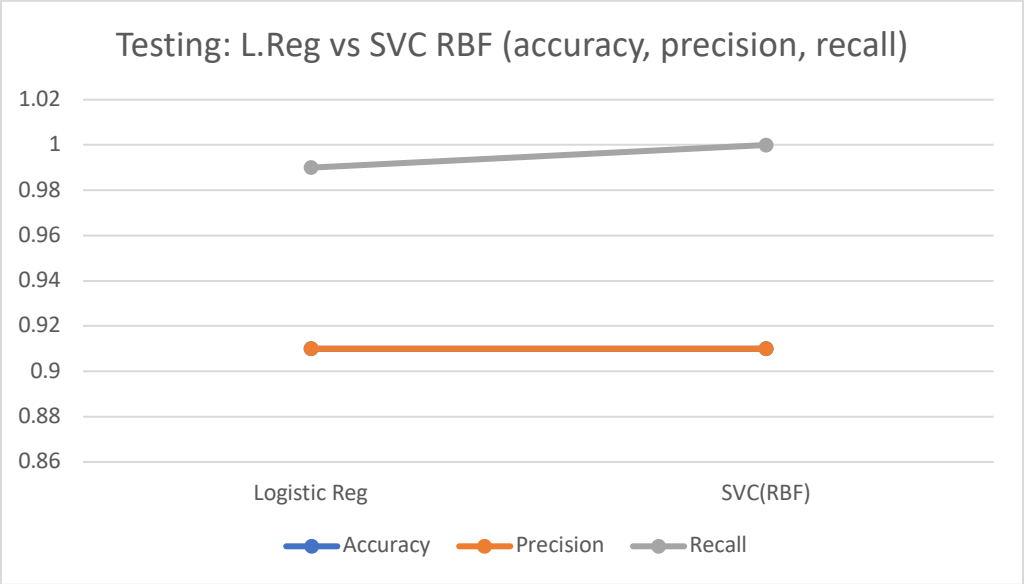
Candidate: Logistic Regression, Logistic regression got recall and accuracy score higher than KNN even though AUC score is less than KNN but it is acceptable for Logistic regression is perform slightly better than KNN base on the problem.

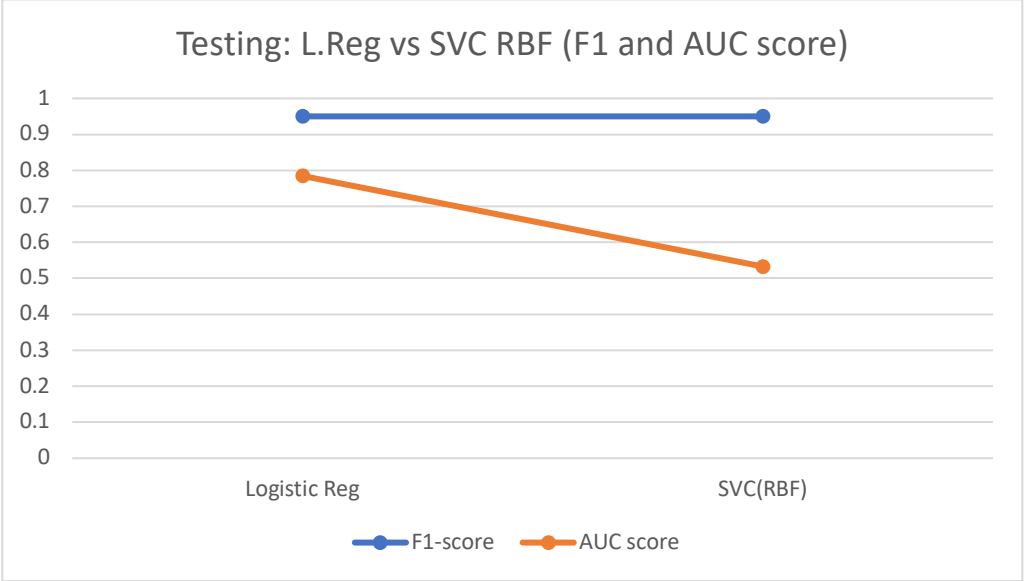




Candidate: SVC(RBF), SVC(RBF) got higher accuracy score compare with SVC(Linear) which has similar result.

Compare testing result between Logistic Regression and SVC(BRF)





Best model for Testing: Logistic Regression, logistic regression,and SVC(RBF) got most similar overall score; SVC(RBF) got higher recall score than KNN for 0.01 but it does perform in AUC score far away from logistic regression which is the big gap. In this case I decide to choose "Logistic regression" as the best model for testing because recall score that is less than KNN for 0.01 is not the big deal.

Summary train and test without hyperparameter tuning

Note that some models have hyperparameter tuning, all are initial value no modification.

Default value of hyperparameter(for model that has)

KNN: n_neighbors = 5

SVC(Poly): degree = 5

Best model for training:

Model	Accuracy	Precision	Recall	F1-score	AUC score
KNN	0.91	0.91	0.99	0.95	0.9181

- **True negative:** 22591
- **True positive:** 589
- **False negative:** 236
- **False positive:** 1977

Best model for testing:

Model	Accuracy	Precision	Recall	F1-score	AUC score
Logistic Reg	0.91	0.91	0.99	0.95	0.7845

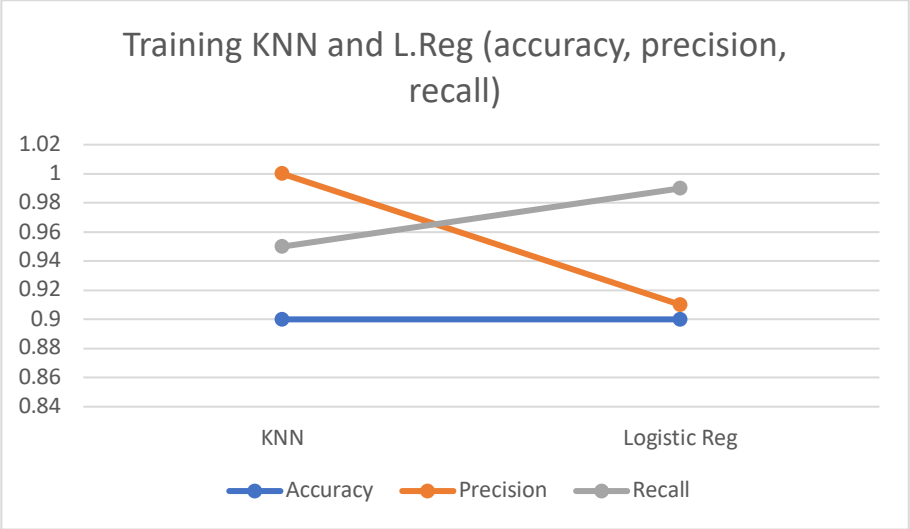
- **True negative:** 5706
- **True positive:** 44
- **False negative:** 40
- **False positive:** 550

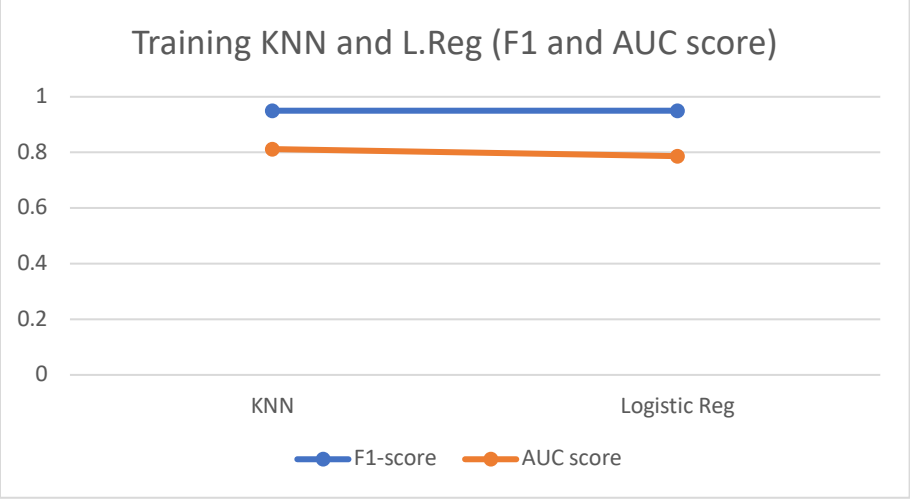
Conclusion: the model that got overall performance for both training and testing(without hyperparameter tuning) is “SVC(RBF)” because this model has acceptable overall score and has less variance compared to the rest models.

Train and Test result with hyperparameter tuning

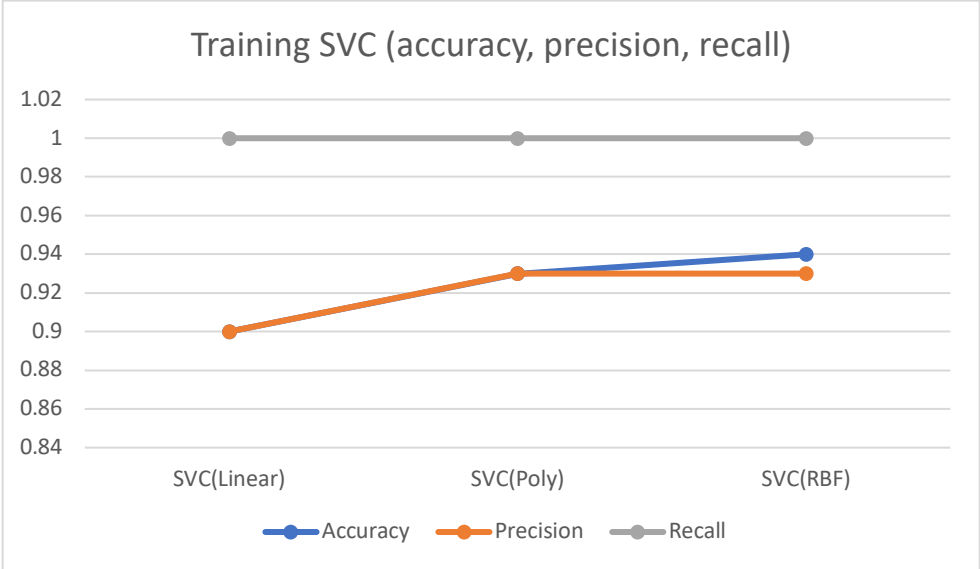
Training

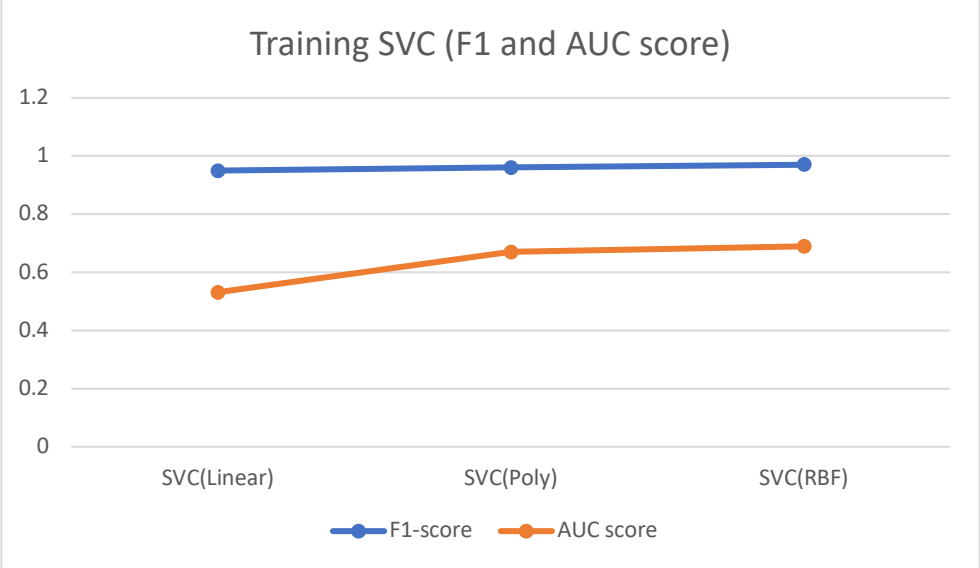
Model	Accuracy	Precision	Recall	F1-score	AUC score	Hyperparameter
KNN	0.90	1.00	0.95	0.95	0.8120	n_neighbors=55
Logistic Reg	0.90	0.91	0.99	0.95	0.7864	penalty='l2', C=1000
SVC(Linear)	0.90	0.90	1.00	0.95	0.5315	C=1
SVC(Poly)	0.93	0.93	1.00	0.96	0.6704	degree=23
SVC(RBF)	0.94	0.93	1.00	0.97	0.6894	C=1000





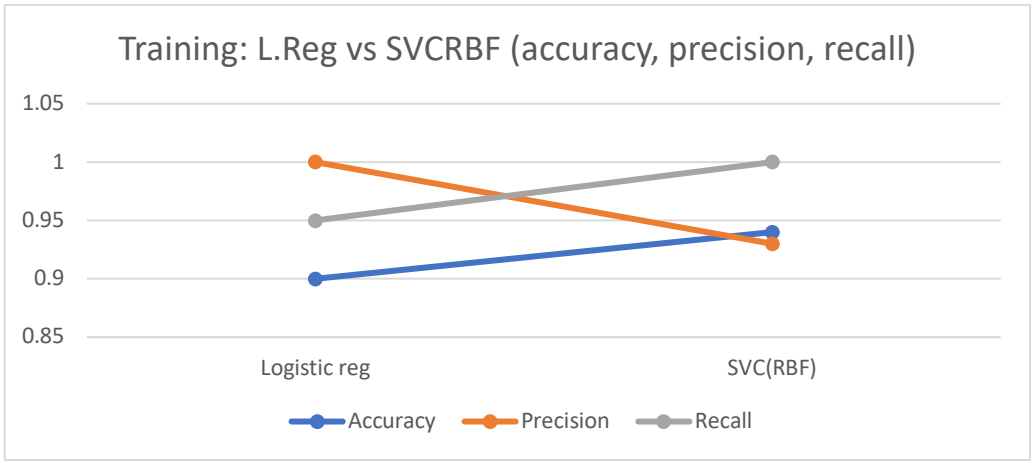
Candidate: Logistic regression, even precision KNN got higher score but Logistic regression got much better in recall which is more important score that I should concern so I choose Logistic regression.

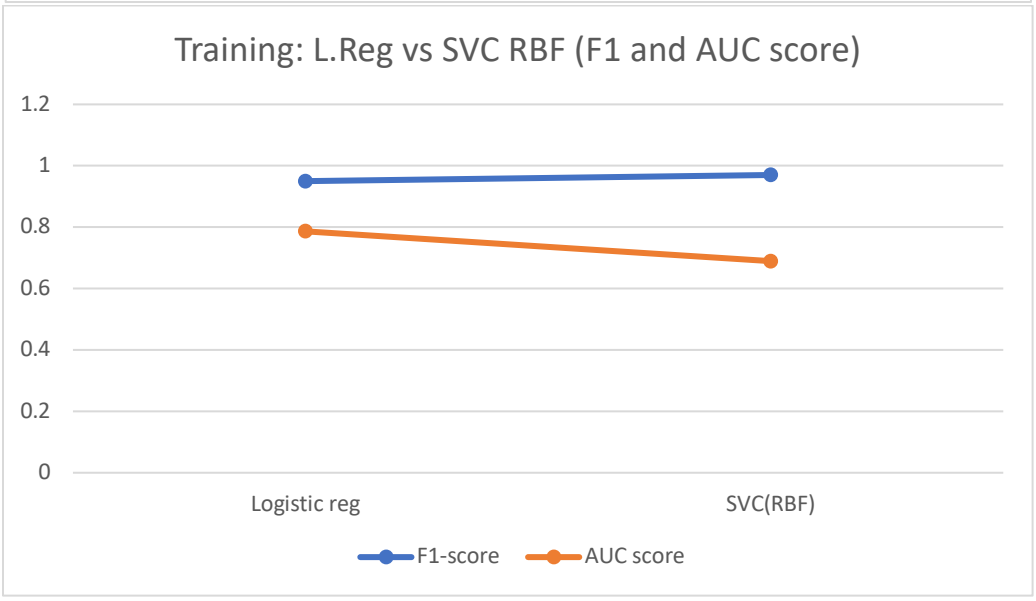




Candidate: SVC(RBF) because this model archives the highest accuracy, F1-score and AUC score compare with SVC linear and polynomial, in this case three of them got the same recall.

Compare training result between Logistic regression and SVC(BRF)

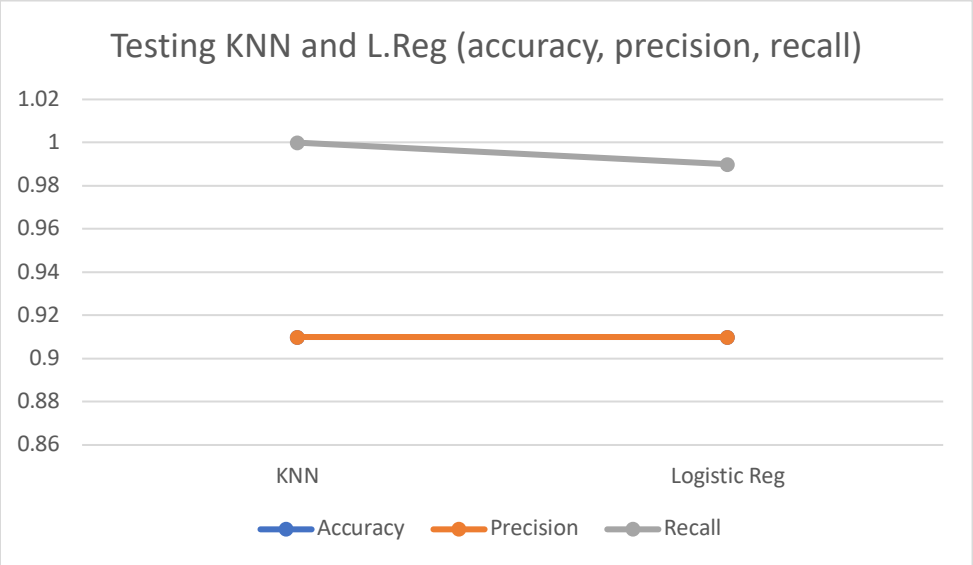


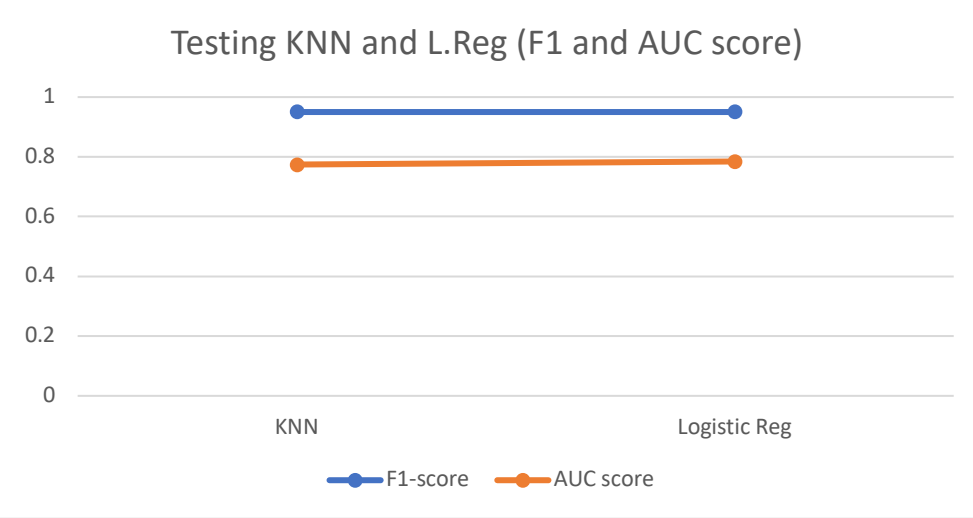


Best model for training: SVC(RBF) because this model got higher recall, accuracy, precision, and F1-score even though it got lower AUC score than Logistic regression at 0.1 which is not the big deal.

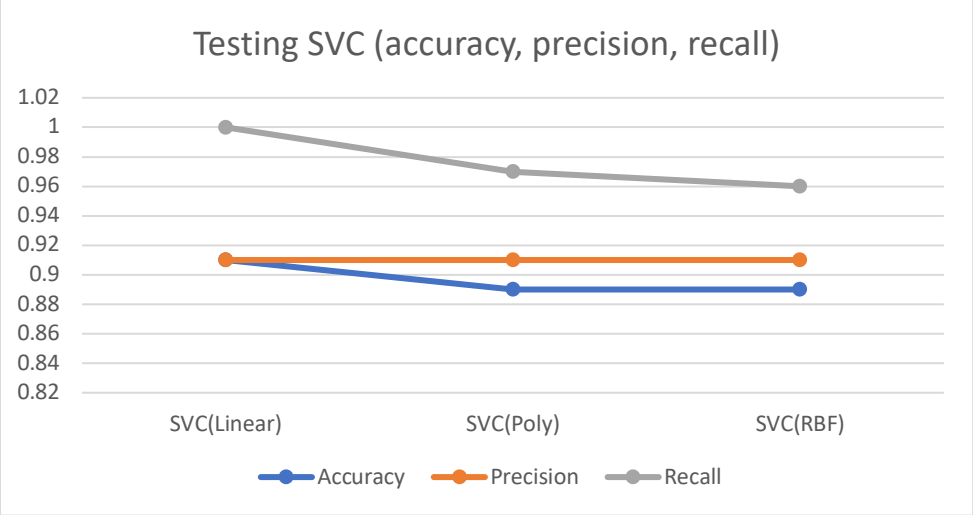
Testing

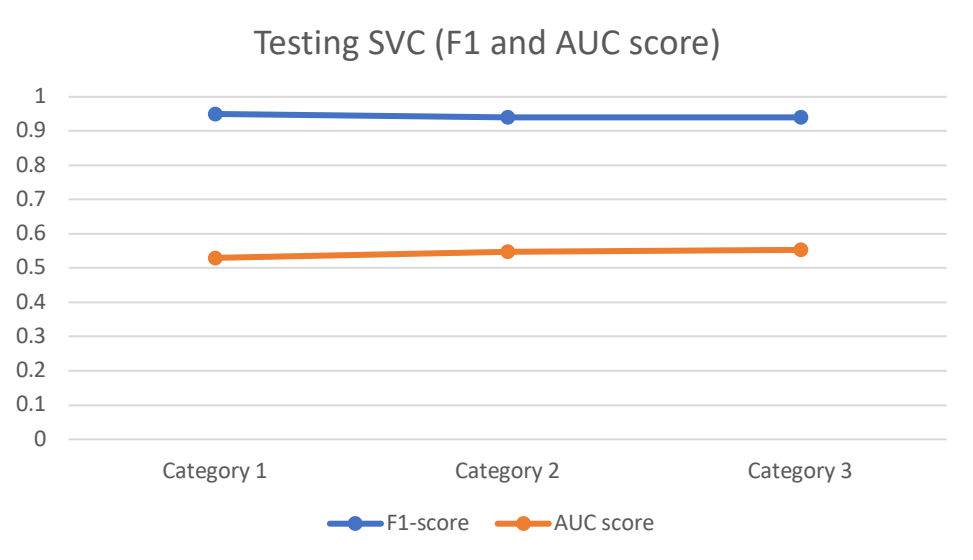
Model	Accuracy	Precision	Recall	F1-score	AUC score	Hyperparameter
KNN	0.91	0.91	1.00	0.95	0.7743	n_neighbors=55
Logistic Reg	0.91	0.91	0.99	0.95	0.7845	penalty='l2', C=1000
SVC(Linear)	0.91	0.91	1.00	0.95	0.5298	C=1
SVC(Poly)	0.89	0.91	0.97	0.94	0.5476	degree=23
SVC(RBF)	0.89	0.91	0.96	0.94	0.5532	C=1000





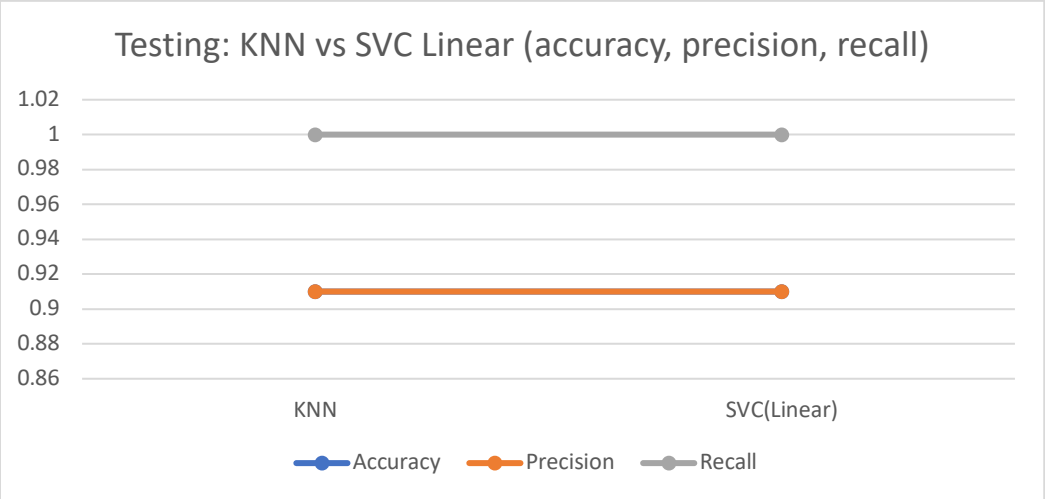
Candidate: KNN from consider of recall first, even though Logistic regression got higher AUC score.

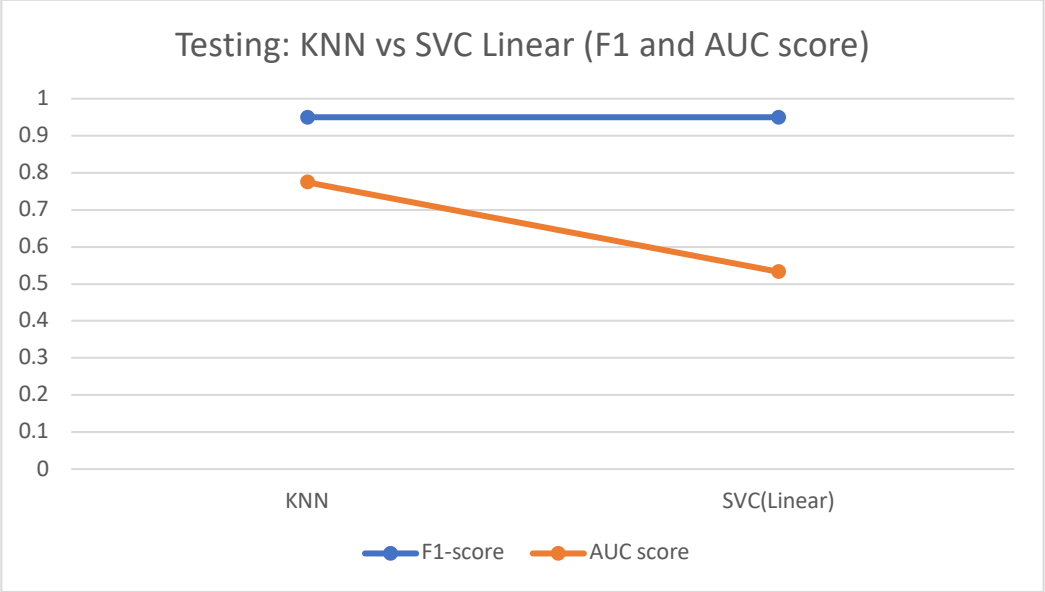




Candidate: SVC(Linear) because this model got more higher score in recall and accuracy which is better than polynomial and RBF even though this model got lower AUC score than those two models for 0.2 to 0.3.

Compare training result between KNN and SVC(Linear)





Best model for testing: KNN, since both of them has very similar result but KNN got better AUC score compare with SVC(Linear) so I choose KNN.

Summary train and test with hyperparameter tuning

Best model for training:

Model	Accuracy	Precision	Recall	F1-score	AUC score	Hyperparameter
SVC(RBF)	0.89	0.91	0.96	0.94	0.5532	C=1000

- **True negative:** 22767
- **True positive:** 979
- **False negative:** 60
- **False positive:** 1597

Best model for testing:

Model	Accuracy	Precision	Recall	F1-score	AUC score	Hyperparameter
KNN	0.91	0.91	1.00	0.95	0.5327	degree=23

- **True negative:** 5732
- **True positive:** 41
- **False negative:** 14
- **False positive:** 562

Best model for training: SVC(Linear) because this model got acceptable overall score with lower variance of training and testing result compare with the rest models.

Appendix

```
[[22744    83]
 [ 2389   177]]
              precision    recall  f1-score   support

           1       0.90      1.00      0.95     22827
           2       0.68      0.07      0.13      2566

    accuracy                           0.90     25393
   macro avg       0.79      0.53      0.54     25393
weighted avg       0.88      0.90      0.87     25393

0.8120747927385358
```

I. KNN training result

```
[[5729   17]
 [ 568   35]]
              precision    recall  f1-score   support

           1       0.91      1.00      0.95      5746
           2       0.67      0.06      0.11       603

    accuracy                           0.91      6349
   macro avg       0.79      0.53      0.53      6349
weighted avg       0.89      0.91      0.87      6349

0.7743483245104099
```

II. KNN testing result

```
[[22654   173]
 [ 2341   225]]
              precision    recall  f1-score   support

           1       0.91      0.99      0.95     22827
           2       0.57      0.09      0.15      2566

    accuracy                           0.90     25393
   macro avg       0.74      0.54      0.55     25393
weighted avg       0.87      0.90      0.87     25393

0.7864167636464195
```

III. Logistic regression training result

```
[[22654   173]
 [ 2341   225]]
              precision    recall  f1-score   support

           1       0.91      0.99      0.95     22827
           2       0.57      0.09      0.15      2566

    accuracy                           0.90     25393
   macro avg       0.74      0.54      0.55     25393
weighted avg       0.87      0.90      0.87     25393

0.7864167636464195
```

IV. Logistic regression testing result

```
[[22718   109]
 [ 2392   174]]
              precision    recall  f1-score   support

           1       0.90      1.00      0.95     22827
           2       0.61      0.07      0.12      2566

    accuracy                           0.90     25393
   macro avg       0.76      0.53      0.53     25393
weighted avg       0.88      0.90      0.86     25393

0.5315173868196518
```

V. SVC(Linear) training result

```
[[5727   19]
 [ 565   38]]
              precision    recall  f1-score   support

           1       0.91      1.00      0.95      5746
           2       0.67      0.06      0.12       603

    accuracy                           0.91      6349
   macro avg       0.79      0.53      0.53      6349
weighted avg       0.89      0.91      0.87      6349

0.5298557970098458
```

VI. SVC(Linear) testing result

```
[[22791    36]
 [ 1687   879]]
              precision    recall  f1-score   support

           1       0.93      1.00      0.96     22827
           2       0.96      0.34      0.51      2566

    accuracy                           0.93     25393
   macro avg       0.95      0.67      0.73     25393
weighted avg       0.93      0.93      0.92     25393

0.6704897142049959
```

VII. SVC(Polynomial) training result

```
[[5732   14]
 [ 562   41]]
              precision    recall  f1-score   support

           1       0.91      1.00      0.95      5746
           2       0.75      0.07      0.12       603

    accuracy                           0.91      6349
   macro avg       0.83      0.53      0.54      6349
weighted avg       0.90      0.91      0.87      6349

0.5327784444756147
```

VIII. SVC(Polynomial) testing result

```
[[22767    60]
 [ 1587   979]]
              precision    recall  f1-score   support

           1       0.93      1.00      0.97     22827
           2       0.94      0.38      0.54      2566

    accuracy                           0.94     25393
   macro avg       0.94      0.69      0.75     25393
weighted avg       0.94      0.94      0.92     25393

0.6894496016173159
```

IX. SVC(RBF) training result

```
[[5538  208]
 [ 517   86]]
              precision    recall  f1-score   support

           1       0.91      0.96      0.94      5746
           2       0.29      0.14      0.19       603

    accuracy                           0.89      6349
   macro avg       0.60      0.55      0.57      6349
weighted avg       0.86      0.89      0.87      6349

0.5532105685749233
```

X. SVC(RBF) testing result