BY GROUP 17

Python programming
in medical data analysis

FINAL PROJECT

B323111023 鄭宇翔
B313111055 洪宣蓉
B313111056 潘湘昀

OUTLINE

INTRODUCTION

Machine learning identifies girls with central precocious puberty based on multisource data

Central precocious puberty(CPP)

A disease caused by premature activation of the hypothalamic-pituitary-gonadal (HPG) axis with clinical pubertal symptoms among girls under 8 years old and boys under 9 years of age.

CPP diagnosis

We usually use gonadotropin-releasing hormone (GnRH) stimulation test ,which is the gold standard for CPP diagnosis.

Objective

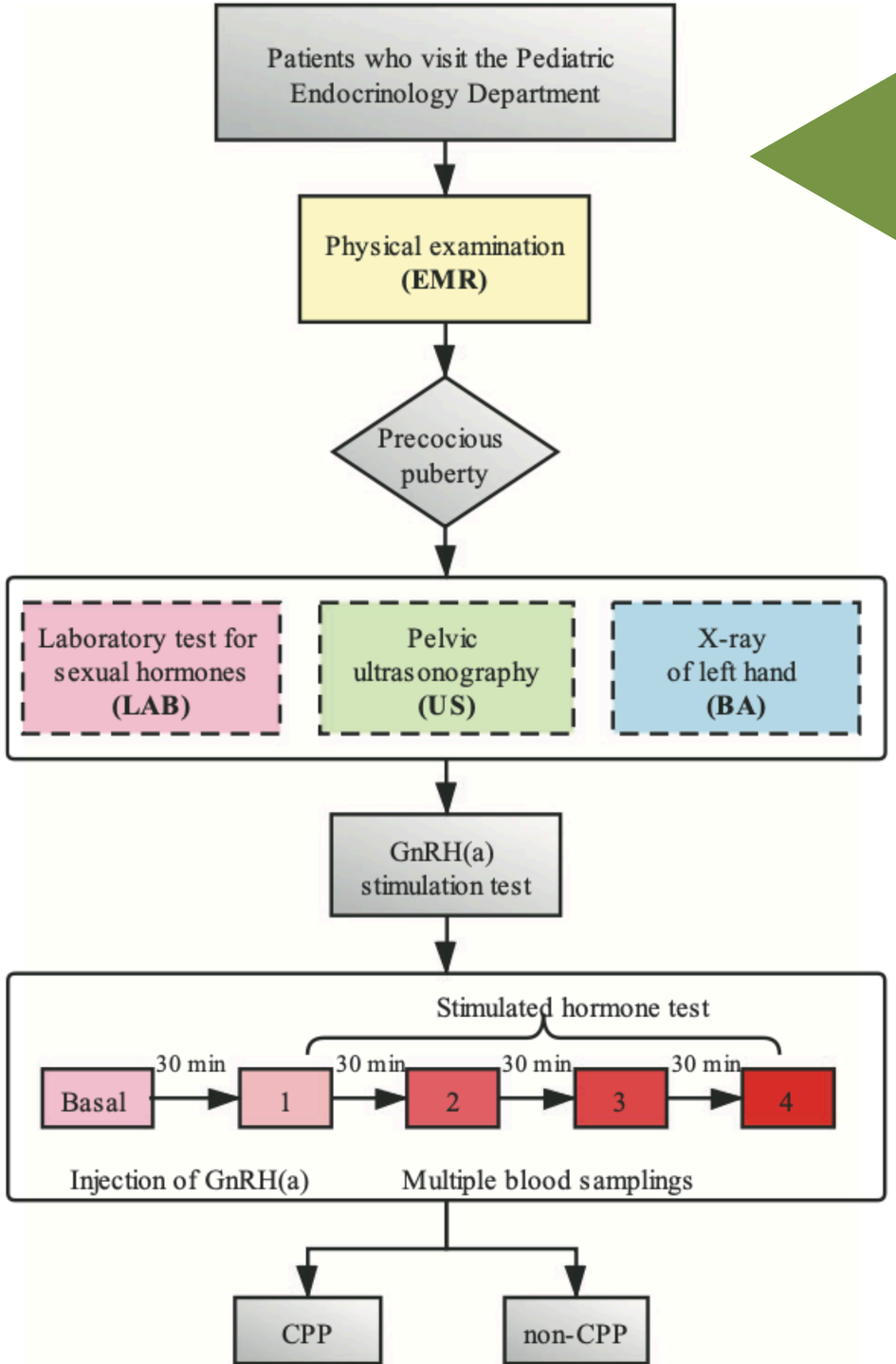
The study aimed to develop simplified diagnostic models for identifying girls with CPP, without GnRH stimulation test.

METHOD

CPP diagnosis pathway

the data obtained using these different approaches from the same patient were considered as data from different sources.

Data sources:EMR, LAB, BA, and US



LAB

Laboratory test for sexual harmones

Python-programming-in-medical-data-analysis / shared_data / test.csv

shtthfkup Add files via upload

Preview | Code | Blame | 2524 lines (2524 loc) · 95.8 KB | Code 55% faster with GitHub Copilot

Search this file

1	LH	FSH	IGF-1	IGFBP-3	GH	TTE	PRL	E2	Age	IS_CPP
2	0.07	0.3	403.0	5.66	3.16	0.35	14.73	66.54	8	0
3	0.07	2.36				0.35	3.96	130.79	5	0
4	0.09	4.04	232.0	3.98	0.29	0.79	6.49	154.86	5	0
5	0.07	1.87	188.0	3.94	2.55	0.35	4.59	60.54	8	0
6	0.07	0.8	356.0	5.48	1.67	0.86	2.99	72.44	8	1
7	0.39	4.24				0.77	14.19	81.45	7	0
8	1.89	2.01	288.0	4.24	9.39	1.66	7.24	83.61	7	1
9	0.12	1.74				1.11	13.31	116.04	7	0
10	0.1	3.23	204.0		7.75	0.35	3.01	121.16	8	0

OUTLINE

-10 variables

(LH,FSH,IGF-1, IGFBP-3,GH,TTE,PRL, E2,Age,IS_CPP)

-2524 rows data

-CPP 1153

Non-CPP 1370

FEATURE	NON-CPP	CPP	P-VALUE
LH (IU/L)	0.11	0.78	< 0.001
FSH (IU/L)	1.83	2.94	< 0.001
IGF-1 (ng/L)	232.25	295.19	< 0.001
IGFBP-3 (μg/mL)	4.69	0.78	< 0.001
GH (ng/mL)	3.18	4.13	< 0.001
TTE (nmol/L)	0.80	0.90	< 0.001
PRL (ng/mL)	9.42	8.765	< 0.001
E2 (ng/mL)	106.03	120.79	< 0.001
AGE (years)	7.05	7.47	< 0.001

TEST_BA

X-ray of left hand

Python-programming-in-medical-data-analysis / shared_data / test_ba.csv

shtthfkup Add files via upload

Preview | Code | Blame | 1613 lines (1613 loc) · 81.8 KB | Code 55% faster with GitHub Copilot

Search this file

1	LH	FSH	IGF-1	IGFBP-3	GH	TTE	PRL	E2	Age	IS_CPP	BoneAge	Age_ratio
2	0.07	0.47	217.0	3.81	6.12		6.83	50.75	8	0	9.0	1.125
3	1.22	3.48	180.0		0.24				7	1	10.0	1.428571429
4	0.85	3.45	421.0			0.35			8	1	11.0	1.375
5	0.07	1.09	312.0		3.02	0.76			7	0	8.0	1.142857143
6	2.13	2.87	429.0	4.78	7.38	0.35			7	1	9.0	1.285714286
7	0.19	2.92	298.0	5.78	5.21		4.97	188.18	8	1	10.5	1.3125
8	0.07	0.97	278.0	4.96	0.09	0.48	6.45	116.07	6	0	9.0	1.5
9	0.94	0.63			3.4	0.35			8	1	11.0	1.375
10	0.07	0.96	238.0	4.72	2.74	0.81	9.75	58.58	6	0	8.25	1.375
11	0.07	0.85	285.0	4.98	3.18	0.92	3.82	81.04	8	1	11.0	1.375
12	0.07	1.19	231.0	5.49	1.91	0.96	11.2	119.28	8	0	8.25	1.03125
13	0.07	1.37	222.0	5.35	1.8	0.35			7	0		1.25

OUTLINE

-2 more variables
(BoneAge, Age_ratio)

-Age_ratio= BoneAge/Age

-1613 rows data

-CPP 715
Non-CPP 897

FEATURE	NON-CPP	CPP	P-VALUE
BoneAge (years)	897	715	< 0.001
Age Ratio	1.25	1.32	< 0.001

TEST_EMR

Physical Examination

Python-programming-in-medical-data-analysis / shared_data / test_emr.csv

shtthfkup Add files via upload

Preview | Code | Blame | 1676 lines (1676 loc) · 114 KB | Code 55% faster with GitHub Copilot

Search this file

	LH	FSH	IGF-1	IGFBP-3	GH	TTE	PRL	E2	Age	IS_CPP	abnormal	height	weight	breast	core	vulva	pubes	pigmentation
2	0.13	1.18			3.35			135.0	7	0	6.0	126.7	25.0					
3	0.26	2.19				0.69		124.0	8	0	6.0	128.3	32.0					
4	0.32	2.55				0.69		259.0	8	0	3.0	126.3	21.0					
5	0.1	1.62				0.69		167.0	6	0	0.2	120.7	22.0					
6	0.1	0.75	341.0		3.31			137.0	7	0	3.0	138.8	38.0					
7	0.1	0.82			0.42	0.69		124.0	8	0	18.0	131.8	26.0					
8	0.1	1.52				0.69		127.0	6	0	3.0	128.9	23.0					
9	1.15	4.19				0.69		157.0	6	1	24.0	119.7	23.5					
10	0.18	1.36				0.69		136.0	8	1	3.0	136.5	34.0					
11	0.73	3.84				0.69		162.0	5	1	3.0	109.3	18.0					
12	0.99	3.55				0.69		232.0	8	1	6.0	142.0	32.0					
13	0.1	1.52				0.69		196.0	7	0	6.0	131.9	25.0					

OUTLINE

- -8 more variables
- -1676 rows data
- -CPP 794
  Non-CPP 881
- -Abnormal
  =abnormal duration
- -Breast formal:
  tanner stage 1-5
- -Vulva/Pubes formal:
  tanner stage 1-3
- -Core/Pigmentation
  formal: Yes/No

FEATURE	NON-CPP	CPP	P-VALUE
Duration (years)	8.15	9.95	< 0.001
Height (cm)	126.11	129.95	< 0.001
Weight (kg)	26.04	28.33	< 0.001
BMI (kg/m^2)	16.22	16.66	< 0.001
Breast tanner stage 1	72	47	< 0.001
stage 2	418	197	
stage 3	347	402	
stage 4	43	143	
stage 5	1	5	

FEATURE	NON-CPP	CPP	P-VALUE
Core YES	724	684	0.027
Core NO	157	110	
Vulva tanner stage 1	833	722	0.009
stage 2	44	61	
stage 3	4	11	
Pubes tanner stage 1	837	773	0.002
stage 2	43	17	
stage 3	1	4	
Pigmentation YES	57	38	0.137
Pigmentation NO	824	756	

TEST_US

Pelvic Ultras Onography

Python-programming-in-medical-data-analysis / shared_data / test_us.csv

shtthfkup Add files via upload

Preview | Code | Blame | 2008 lines (2008 loc) · 160 KB | Code 55% faster with GitHub Copilot

Search this file

	LH	FSH	IGF-1	IGFBP-3	GH	TTE	PRL	E2	Age	IS_CPP	uterus1	uterus2	uterus3	lovary1	lovary2	lovary3	rovary1	rovary2	rovary3
2	0.07	0.47	217.0	3.81	6.12		6.83	50.75	8	0	23.0	19.0	13.0	33.0	10.0	14.0	35.0	9.0	15.0
3	1.22	3.48	180.0		0.24				7	1	32.0	22.0	18.0	25.0	18.0	12.0	30.0	19.0	14.0
4	0.85	3.45	421.0			0.35			8	1	24.0	20.0	12.0	27.0	14.0	25.0	13.0		
5	0.07	1.09	312.0		3.02	0.76			7	0	15.0	14.0	10.0	19.0	9.0	10.0	20.0	11.0	11.0
6	0.07	1.74	180.0		2.43	0.53			6	0	20.0	9.0	10.0	26.0	15.0	15.0	19.0	11.0	13.0
7	2.13	2.87	429.0	4.78	7.38	0.35			7	1	31.0	19.0	14.0	16.0	24.0	15.0	26.0	11.0	25.0
8	0.19	2.92	298.0	5.78	5.21		4.97	188.18	8	1	20.5	13.0	9.3	26.0	15.0	14.0	16.0	11.0	10.5
9	0.07	0.97	278.0	4.96	0.09	0.48	6.45	116.07	6	0	21.0	7.0	13.0	25.0	13.0	15.0	23.0	9.0	11.0
10	0.94	0.63			3.4	0.35			8	1	12.0	20.0	17.0	18.0	14.0	15.0	17.0	13.0	16.0
11	0.07	0.96	238.0	4.72	2.74	0.81	9.75	58.58	6	0	17.0	6.0	4.0	23.0	12.0	18.0	25.0	12.0	21.0
12	0.07	0.85	285.0	4.98	3.18	0.92	3.82	81.04	8	1	19.0	9.0	14.0	25.0	13.0	18.0	20.0	11.0	15.0
13	0.07	1.19	231.0	5.49	1.91	0.96	11.2	119.28	8	0	15.0	8.0	13.0	16.0	10.0	17.0	23.0	11.0	20.0
14	0.07	1.37	222.0	5.35	1.8	0.35			7	0	18.0	8.7	5.0	38.0	13.0	7.0	11.8	8.3	5.3
15	0.07	1.1	299.0	5.1	0.62	0.69			7	0	26.0	15.0	10.0	26.0	10.0	11.0	34.0	10.0	11.0

OUTLINE

-9 more variables

-2000 rows data

-CPP 923
Non-CPP 1084

-uterine volume
=uterus1 x uterus2 x uterus3
(length x wideth x height)

-(L)ovary volume
=lovary1 x lovary2 x lovary3
(length x wideth x height)

-(R)ovary volume
=rovary1 x rovary2 x rovary3
(length x wideth x height)

```
merged_df['uterus_vol'] = merged_df['uterus1'] * merged_df['uterus2'] * merged_df['uterus3']
merged_df['lovary_vol'] = merged_df['lovary1'] * merged_df['lovary2'] * merged_df['lovary3']
merged_df['rovary_vol'] = merged_df['rovary1'] * merged_df['rovary2'] * merged_df['rovary3']
```

FEATURE	NON-CPP	CPP	P-VALUE
Uterine Volume (mL)	1.57	2.69	< 0.001
Left Ovarian Volume (mL)	2.04	2.46	< 0.001
Right Ovarian Volume (mL)	1.94	2.29	< 0.001

ANALYSIS

DATA PROCESSING

-Take the logarithm of 8 variables
(LH,FSH,IGF-1,GH,TTE,PRL,E2)

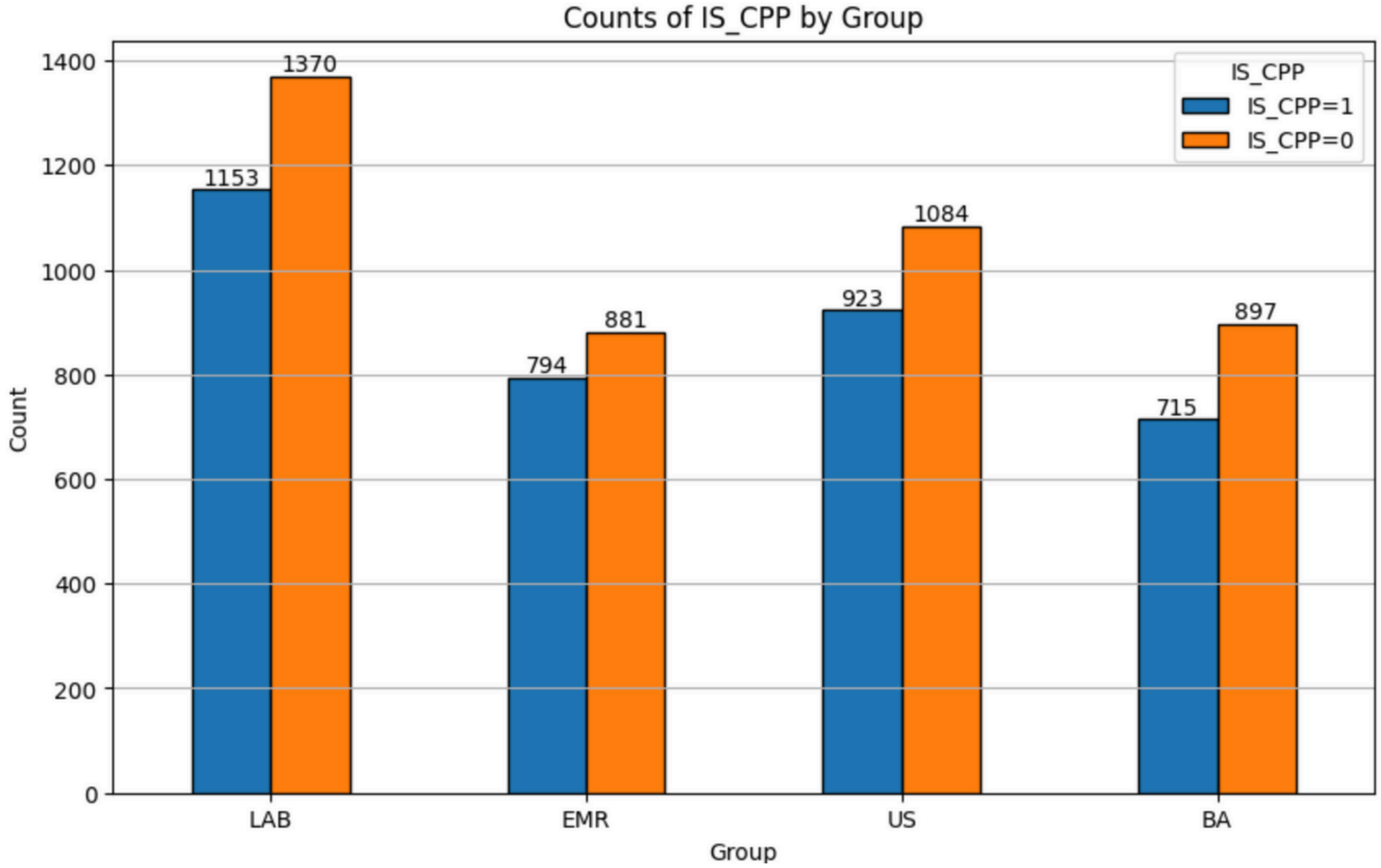
-Get BMI,Bone/Age difference, Uterus/(L)Ovary/(R)Ovary Volume

```
[17]  1 merged_df['LH_log'] = merged_df['LH'].apply(np.log)
      2 merged_df['FSH_log'] = merged_df['FSH'].apply(np.log)
      3 merged_df['IGF-1_log'] = merged_df['IGF-1'].apply(np.log)
      4 merged_df['GH_log'] = merged_df['GH'].apply(np.log)
      5 merged_df['TTE_log'] = merged_df['TTE'].apply(np.log)
      6 merged_df['PRL_log'] = merged_df['PRL'].apply(np.log)
      7 merged_df['E2_log'] = merged_df['E2'].apply(np.log)
```

```
[18]  1 merged_df['BMI'] = merged_df['height']**2/merged_df['weight']
      2 merged_df['Bone_age_diff'] = merged_df['BoneAge'] - merged_df['Age']
      3 merged_df['uterus_vol'] = merged_df['uterus1'] * merged_df['uterus2'] * merged_df['uterus3']
      4 merged_df['lovary_vol'] = merged_df['lovary1'] * merged_df['lovary2'] * merged_df['lovary3']
      5 merged_df['rovary_vol'] = merged_df['rovary1'] * merged_df['rovary2'] * merged_df['rovary3']
```

ANALYSIS

COUNTS OF IS_CPP



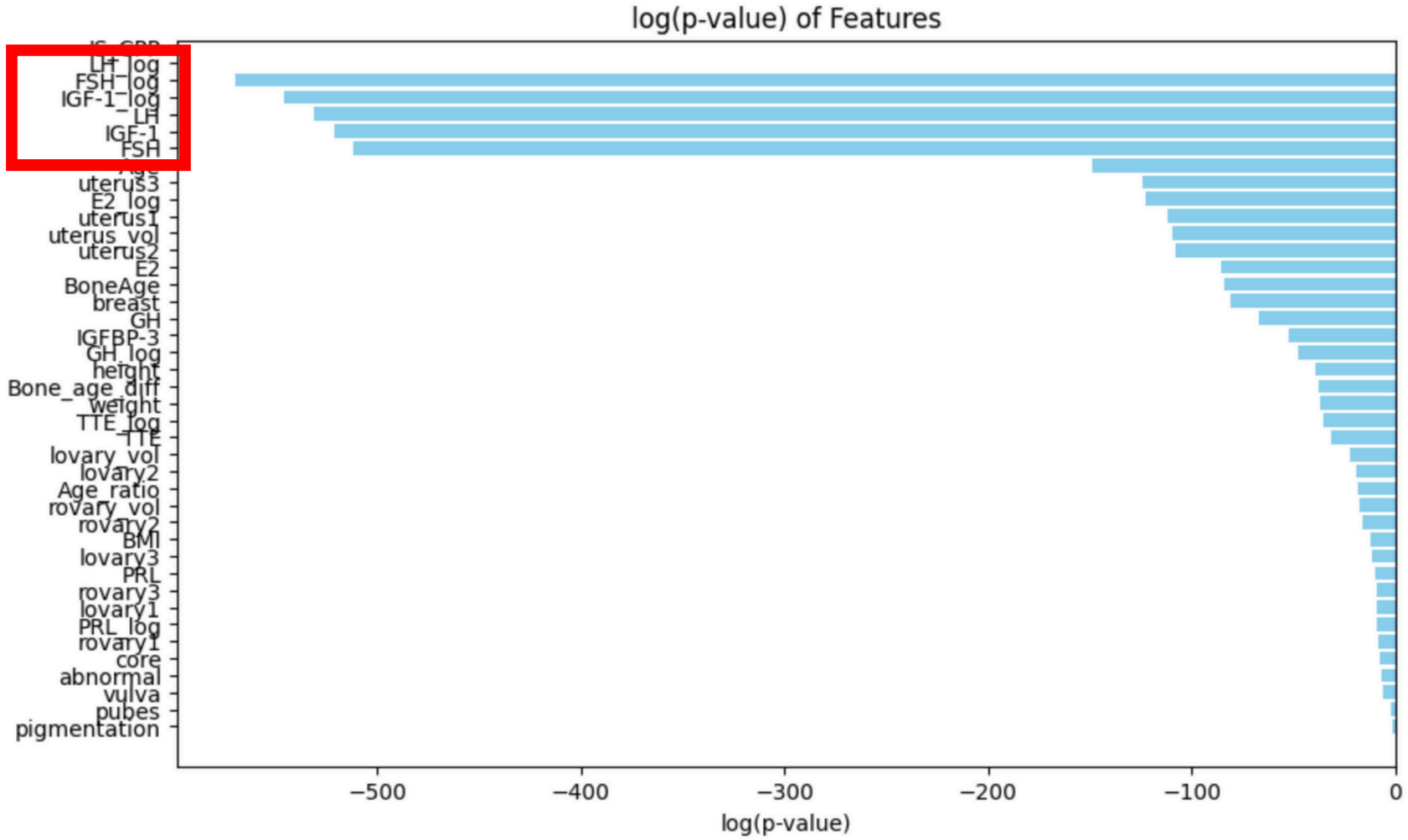
ANALYSIS

COUNTS OF IS_CPP

[45]
```python
import pandas as pd
import matplotlib.pyplot as plt

counts = {
    "LAB": [len(df_LAB[df_LAB["IS_CPP"] == 1]), len(df_LAB[df_LAB["IS_CPP"] == 0])],
    "EMR": [len(df_EMR[df_EMR["IS_CPP"] == 1]), len(df_EMR[df_EMR["IS_CPP"] == 0])],
    "US": [len(df_US[df_US["IS_CPP"] == 1]), len(df_US[df_US["IS_CPP"] == 0])],
    "BA": [len(df_BA[df_BA["IS_CPP"] == 1]), len(df_BA[df_BA["IS_CPP"] == 0])]
}


counts_df = pd.DataFrame(counts, index=["IS_CPP=1", "IS_CPP=0"])
counts_df = counts_df.T

ax = counts_df.plot(kind='bar', figsize=(10, 6), edgecolor='black', rot=0)
plt.title('Counts of IS_CPP by Group')
plt.xlabel('Group')
plt.ylabel('Count')
plt.legend(title='IS_CPP')
plt.grid(axis='y')

for container in ax.containers:
    ax.bar_label(container, label_type='edge')

plt.show()
```

ANALYSIS

LOG(P-VALUE) OF FEATURES



DATA PROCESSING

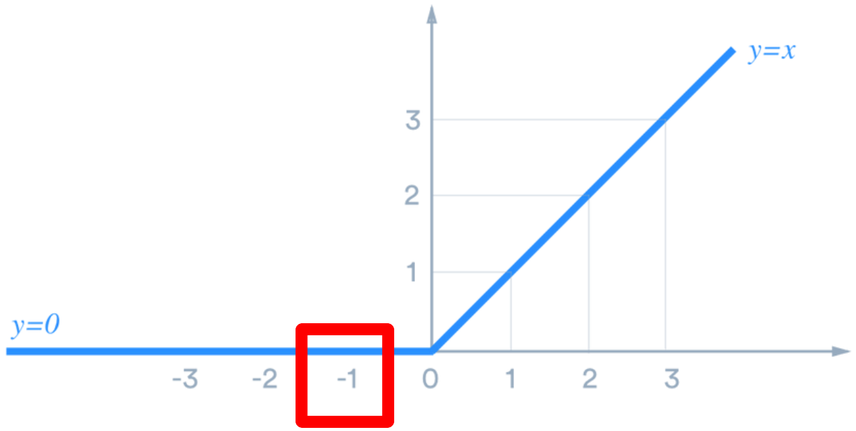
Min Max Normalize (0~1)

```
[19]  1 numeric_cols = merged_df.select_dtypes(include=np.number)
      2 min_vals = numeric_cols.min()
      3 max_vals = numeric_cols.max()
      4 normalized_df = (numeric_cols - min_vals) / (max_vals - min_vals)
      5 normalized_df = pd.concat([normalized_df, merged_df["data_class"]], axis=1)
```

Drop NaN

```
[20]  1 df = normalized_df.fillna(-1)
      2 df
```

To fit Relu Activation



MODEL SVM/RNN

SVM of Age to LH_log

Accuracy: 0.80

RNN of Exclusive data

Batch size = 8

Epoch = 150

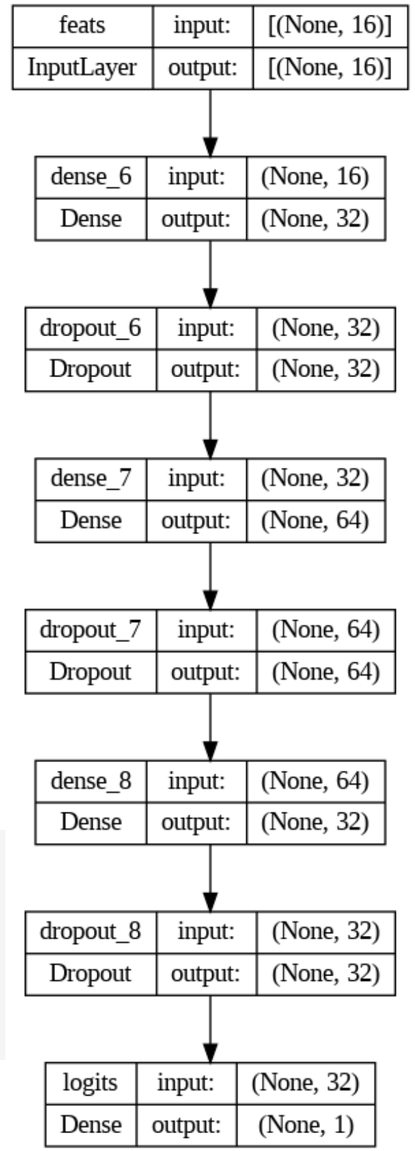
Valid. Data = 0.2

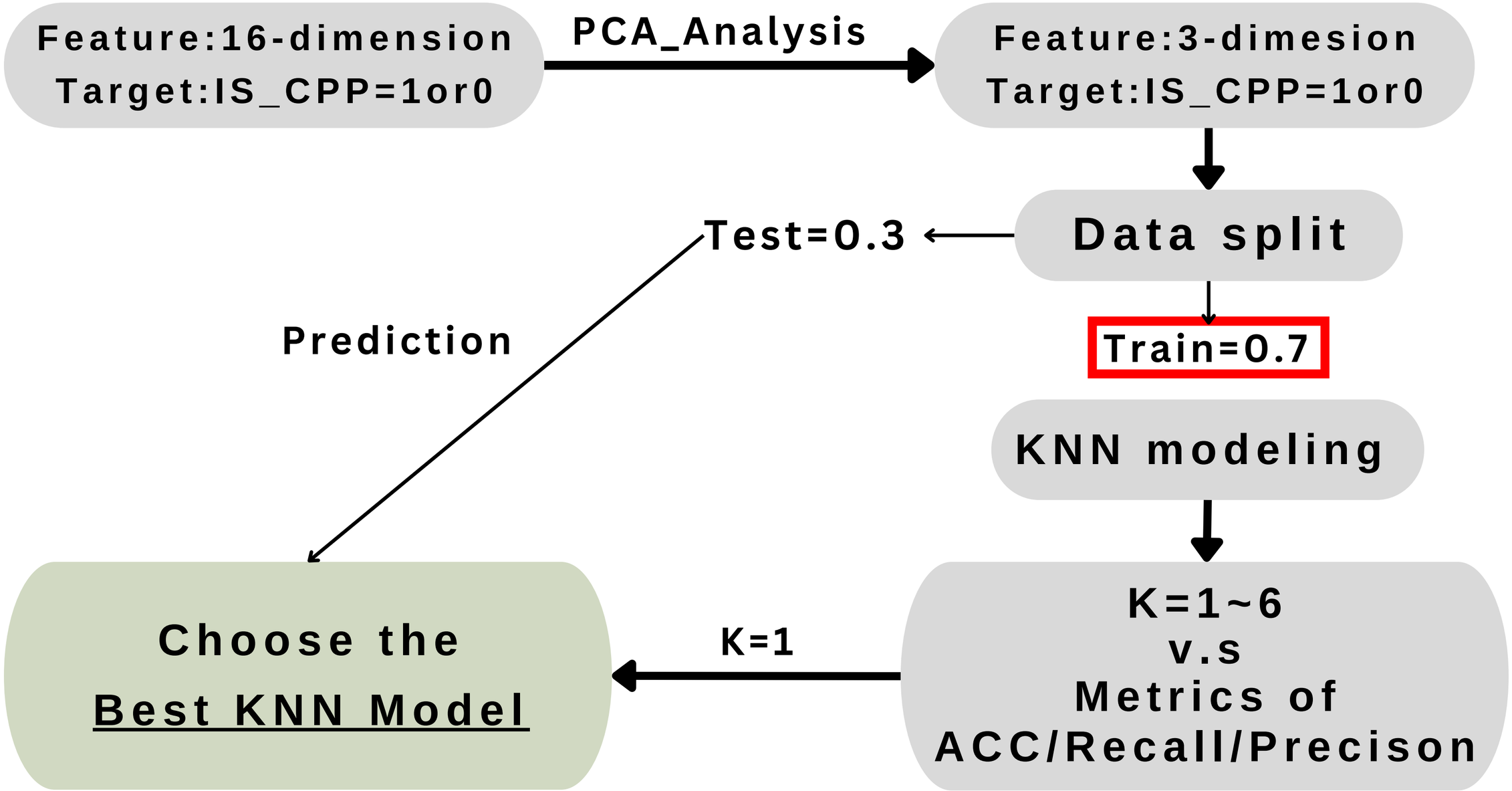
EMR Accuracy: 0.71

BA Accuracy: 0.84

US Accuracy: 0.81

```
inputs = layers.Input(shape=(len(feature_columns), ), name="feats")
x = layers.Dense(32, activation="relu")(inputs)
x = layers.Dropout(0.1)(x)
x = layers.Dense(64, activation="relu")(x)
x = layers.Dropout(0.1)(x)
x = layers.Dense(32, activation="sigmoid")(inputs)
logits = layers.Dense(1, name="logits")(x)
```




Feature:16-dimension
Target:IS_CPP=1or0
PCA_Analysis
Feature:3-dimesion
Target:IS_CPP=1or0
Data split
Test=0.3
Train=0.7
Prediction
KNN modeling
K=1~6
v.s
Metrics of
ACC/Recall/Precison
K=1
Choose the
Best KNN Model

PCA->KNN

Feature:16-dimension
Target:IS_CPP=1or0

PCA_Analysis

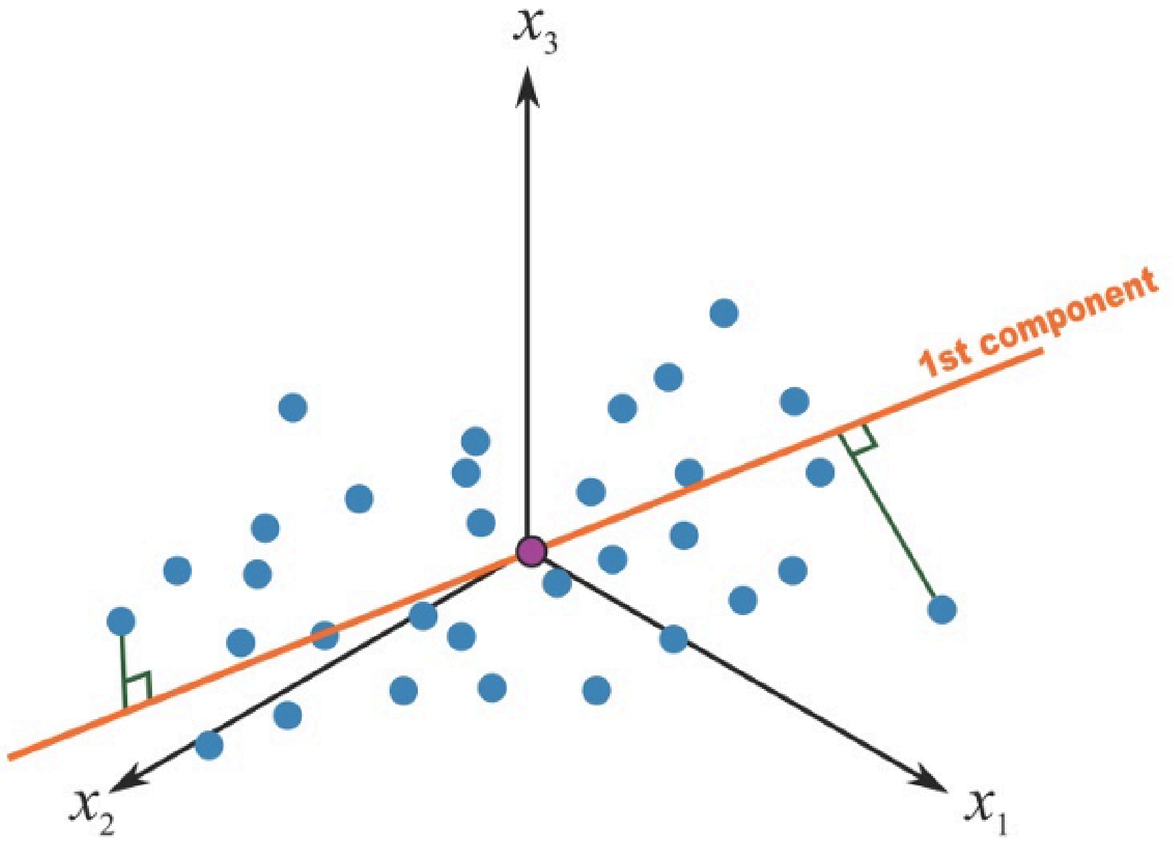
Feature:3-dimesion
Target:IS_CPP=1or0

```
feature_columns = [
    "LH", "FSH", "IGF-1", "IGFBP-3", "GH", "TTE", "PRL", "E2", "Age",
    "LH_log", "FSH_log", "IGF-1_log",
    "GH_log", "TTE_log", "PRL_log", "E2_log"]
```

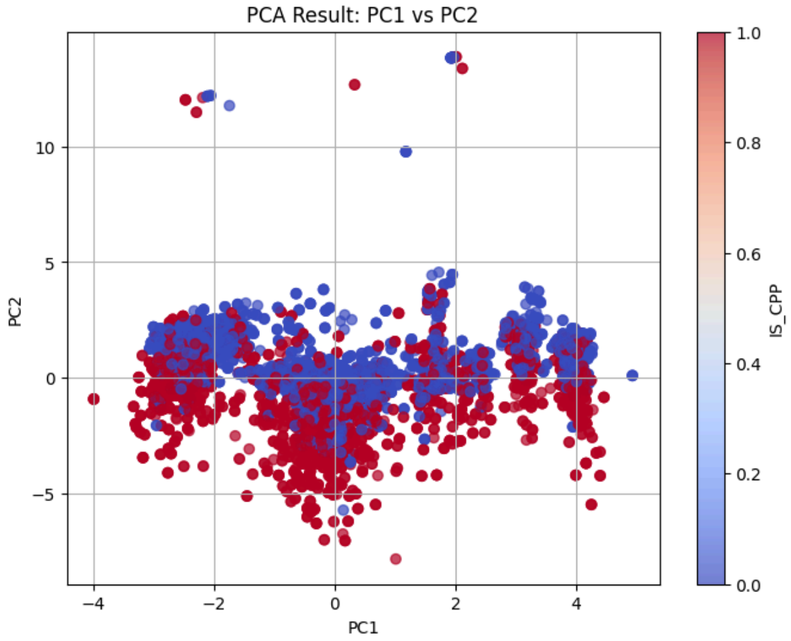
```
pca = PCA(n_components=3)  #3-dimension
pca_result = pca.fit_transform(X_scaled)

pca_df = pd.DataFrame(data=pca_result, columns=['PC1', 'PC2', 'PC3'])
```

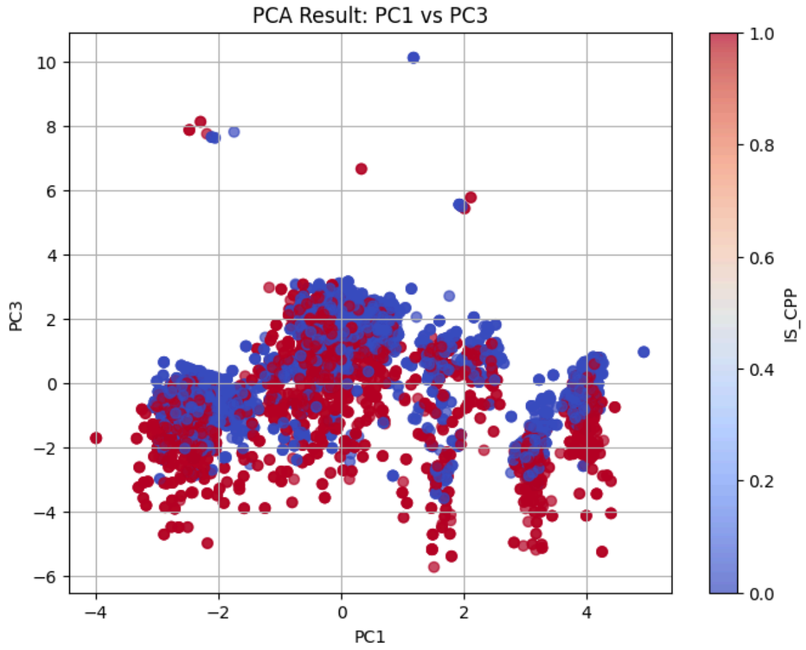
PCA

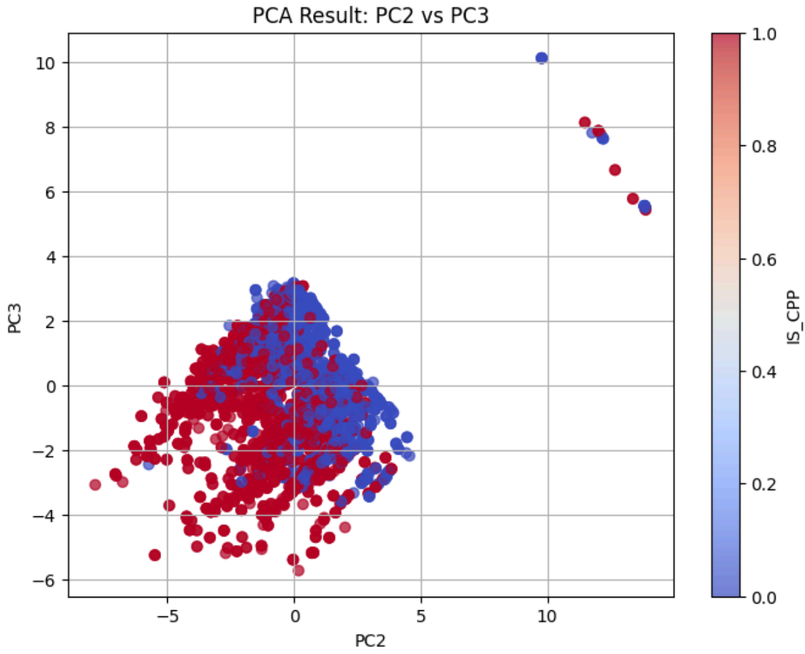
a technique used to reduce the dimensionality of large datasets while retaining most of the variance in the data to reduce complexity and improve computational efficiency.



PCA RESUILT







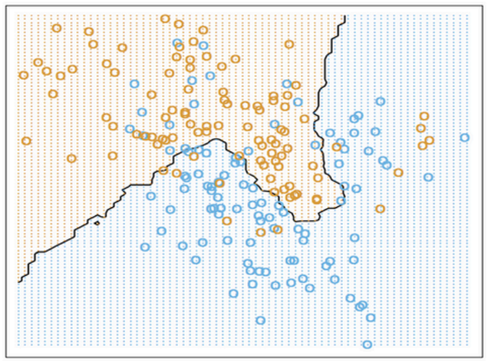
red spot: column of CPP
blue spot :column of non-CPP

DATA SPLITING

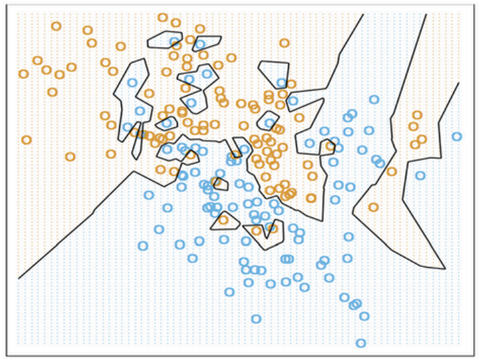
```
pca_df
X = pca_df.drop('IS_CPP', axis=1)
y = pca_df['IS_CPP']

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3, random_state=42)

accuracies = []
recalls = []
precisions = []

# 不同的 K 值
for k in range(1,6):
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)

    y_pred = knn.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred)

    accuracies.append(accuracy)
    recalls.append(recall)
    precisions.append(precision)
```

Train=0.7
Test=0.3

KNN MODELING



15-nearest neighbors

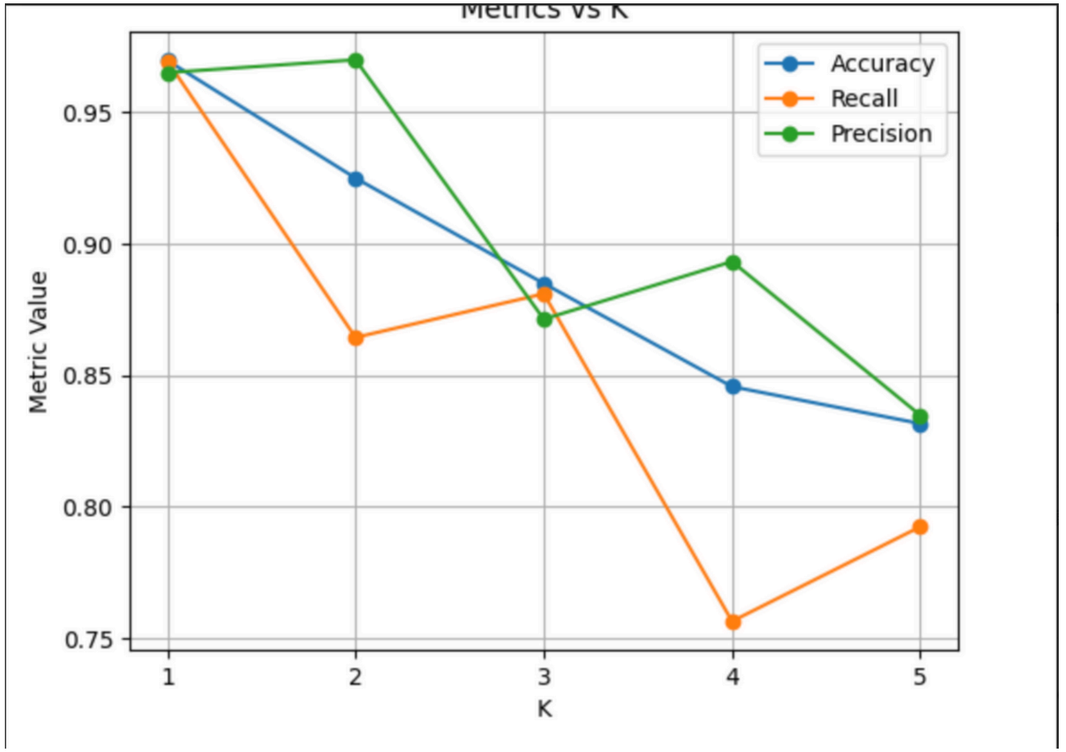


1-nearest neighbors

```
pca_df
X = pca_df.drop('IS_CPP', axis=1)
y = pca_df['IS_CPP']


X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3, random_state=42)

accuracies = []
recalls = []
precisions = []

# 不同的 K 值
for k in range(1,6):
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)

    y_pred = knn.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred)

    accuracies.append(accuracy)
    recalls.append(recall)
    precisions.append(precision)
```

KNN PREDICTION

```python
pca_df
X = pca_df.drop('IS_CPP', axis=1)
y = pca_df['IS_CPP']


X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3, random_state=42)

accuracies = []
recalls = []
precisions = []

# 不同的 K 值
for k in range(1,6):
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)

    y_pred = knn.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred)

    accuracies.append(accuracy)
    recalls.append(recall)
    precisions.append(precision)
```




```python
for k, (accuracy, recall, precision) in enumerate(zip(accuracies, recalls, precisions), 1):
    print(f"K={k}: Accuracy = {accuracy:.4f}, Recall = {recall:.4f}, Precision = {precision:.4f}")
```

```
K=1: Accuracy = 0.9697, Recall = 0.9696, Precision = 0.9651
K=2: Accuracy = 0.9250, Recall = 0.8644, Precision = 0.9700
K=3: Accuracy = 0.8849, Recall = 0.8810, Precision = 0.8714
K=4: Accuracy = 0.8457, Recall = 0.7565, Precision = 0.8932
K=5: Accuracy = 0.8316, Recall = 0.7924, Precision = 0.8348
```

Best model: k=1

acc=0.9697

recall=0.9696

precision=0.9651

BY GROUP 17

THANK YOU

For Your Attention