Technical Report Machine Learning

Explore the Data using Decision Tree, Random Forest, Self-Training use Breast Cancer Dataset and Visualize Data Trends Using Seaborn

Oleh:

Muhammad Althaf Dhiaulhaq (1103194046)

PRODI S1 TEKNIK KOMPUTER
FAKULTAS TEKNIK ELEKTRO
UNIVERSITAS TELKOM
BANDUNG
2023

Pendahuluan

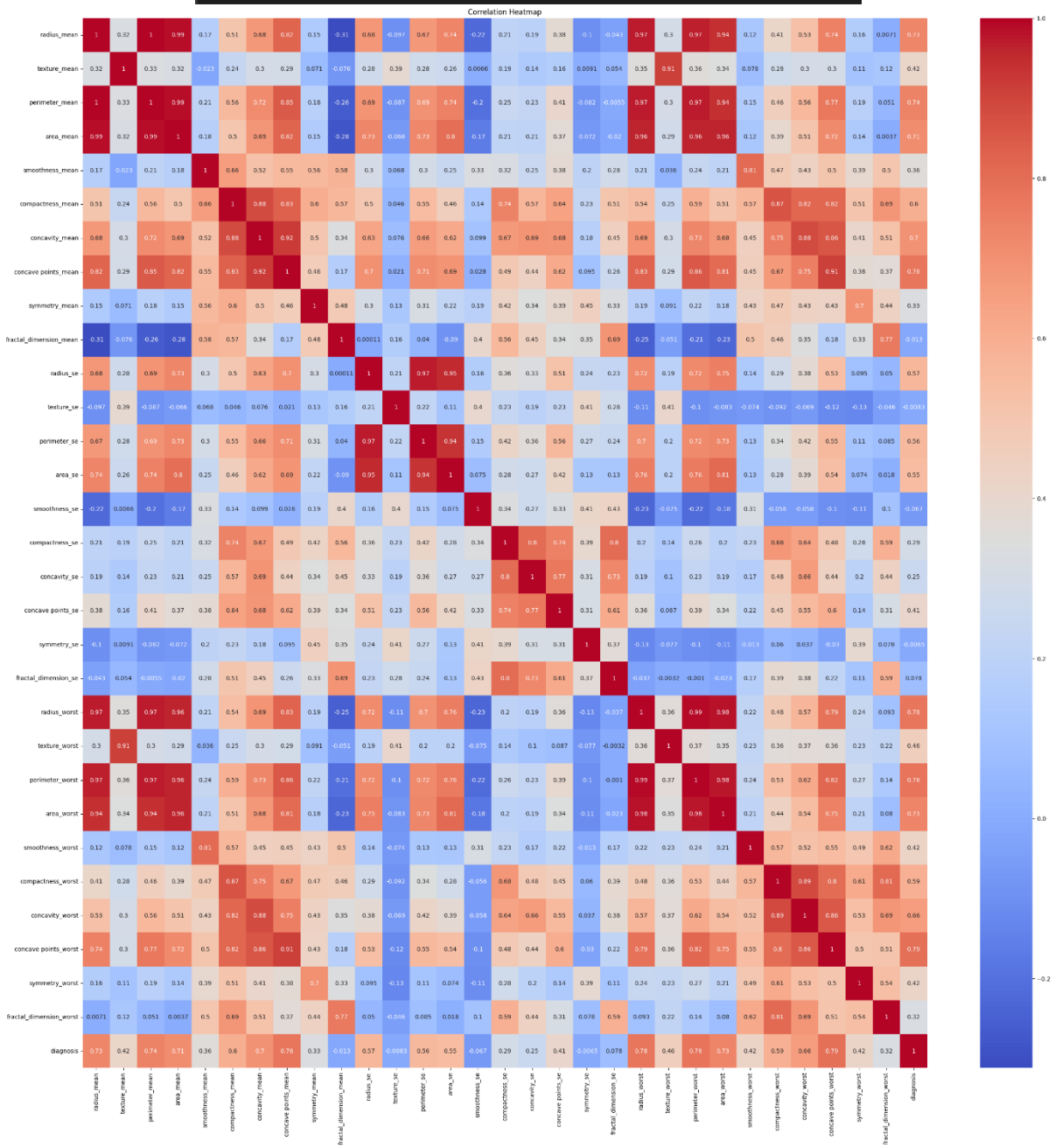
Kanker payudara adalah salah satu bentuk kanker yang paling umum di kalangan wanita. Penting untuk mendeteksi kanker payudara secara dini agar pengobatan yang efektif dapat dilakukan. Salah satu cara untuk mencapai ini adalah dengan menggunakan teknik pembelajaran mesin untuk menganalisis data terkait kanker payudara. Dalam laporan teknis ini, kami akan menjelajahi berbagai teknik, termasuk penggunaan Seaborn, decision tree, random forest, dan pelatihan mandiri menggunakan dataset kanker payudara.

Dataset Kanker Payudara Wisconsin berisi informasi tentang 569 pasien kanker payudara. Dataset ini mencakup 30 fitur yang menggambarkan karakteristik sel yang ditemukan dalam massa payudara. Variabel target dalam dataset ini adalah biner, yaitu ganas atau jinak. Sebelumnya, dataset telah diproses dan tidak mengandung nilai yang hilang.

1. Seaborn

Seaborn adalah sebuah perpustakaan visualisasi data yang dibangun di atas matplotlib dengan antarmuka tingkat tinggi. Perpustakaan ini tidak hanya memungkinkan pembuatan grafik statistik yang informatif dan menarik, tetapi juga mampu memplot berbagai jenis grafik seperti plot pencar, plot garis, plot batang, histogram, dan peta panas. Dalam konteks dataset kanker payudara, Seaborn dapat digunakan untuk memplot matriks korelasi antara variabel yang berbeda, plot kotak untuk membandingkan berbagai kategori, dan plot biola untuk memvisualisasikan distribusi data dengan lebih mudah dan efisien.

```
# Visualisasi dateset using seaborn
plt.figure(figsize=(32,32))
sns.heatmap(datas.corr(), cmap='coolwarm', annot=True)
plt.title('Correlation Heatmap')
plt.show()
```

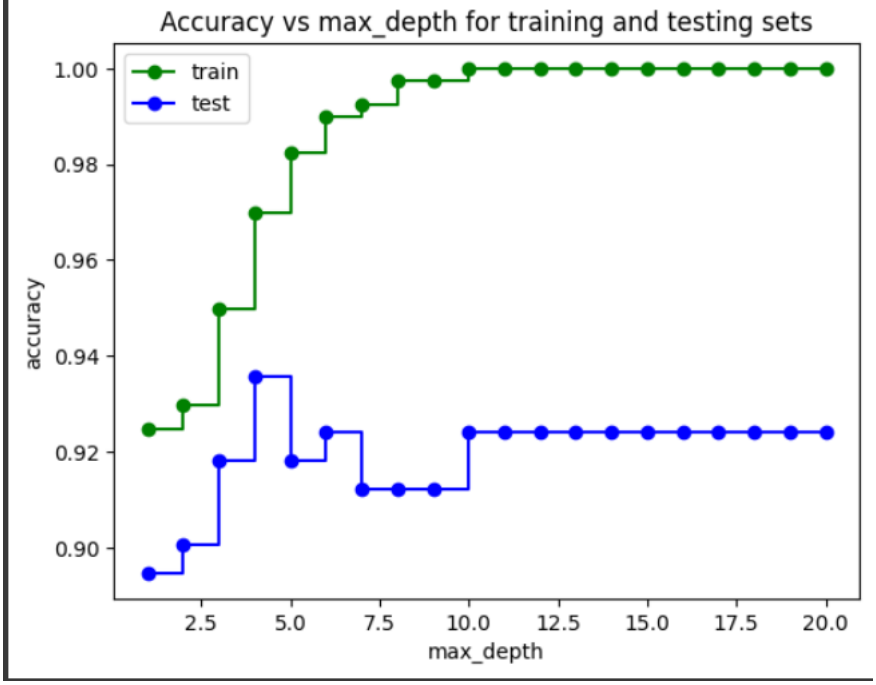


2. Decision Tree

Algoritma Decision Tree merupakan salah satu algoritma pembelajaran terawasi yang digunakan untuk klasifikasi dan regresi. Algoritma ini bekerja dengan membagi data secara rekursif ke dalam subset berdasarkan nilai atribut tunggal pada setiap simpul, dengan tujuan untuk memaksimalkan keuntungan informasi. Dalam analisis ini, kami menggunakan algoritma Decision Tree untuk membangun model klasifikasi pada dataset Kanker Payudara. Dataset tersebut dibagi menjadi data pelatihan dan data uji, dan model Decision Tree dilatih pada data pelatihan. Selanjutnya, kami menggunakan model tersebut untuk memprediksi label kelas pada data uji, dan mengukur akurasi model dengan menggunakan matriks kebingungan.

```
#DECISION TREE
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split

predictors = datas.columns[2:11]
target = "diagnosis"
X = datas.loc[:, predictors]
y = datas[target]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
clf = DecisionTreeClassifier(random_state=42)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
acc = accuracy_score(y_test, y_pred)
print("Accuracy %s" % round(acc*100,2))
depth_range = range(1, 21)
train_scores = []
test_scores = []
for depth in depth_range:
    clf = DecisionTreeClassifier(max_depth=depth, random_state=42)
    clf.fit(X_train, y_train)
    train_scores.append(clf.score(X_train, y_train))
    test_scores.append(clf.score(X_test, y_test))

fig, ax = plt.subplots()
ax.set_xlabel("max_depth")
ax.set_ylabel("accuracy")
ax.set_title("Accuracy vs max_depth for training and testing sets")
ax.plot(depth_range, train_scores, marker="o", label="train", drawstyle="steps-post",color='green')
ax.plot(depth_range, test_scores, marker="o", label="test", drawstyle="steps-post",color='blue')
ax.legend()
plt.show()
```

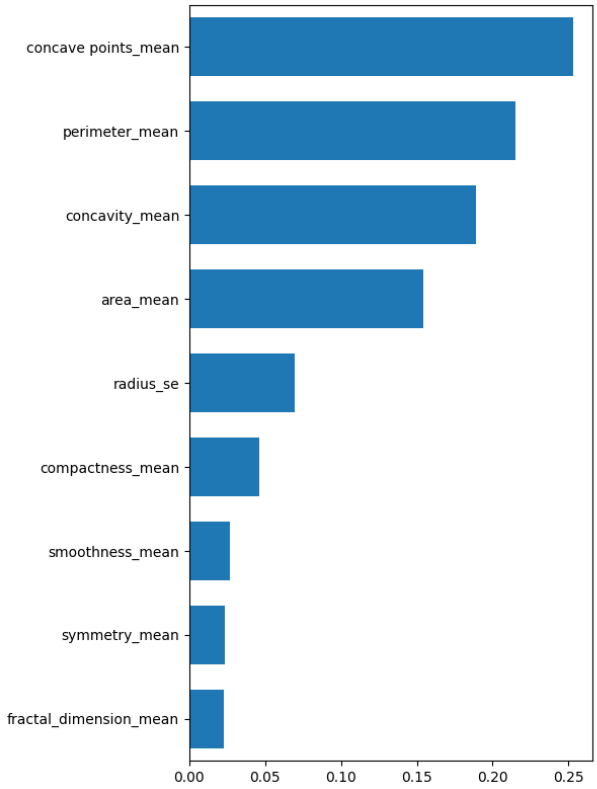
Accuracy 92.4

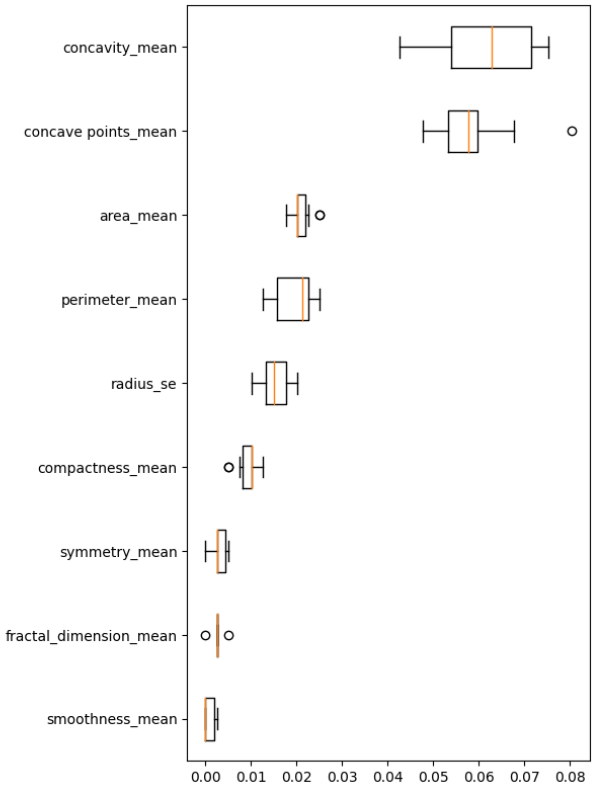


3. Random Forest

Random Forest merupakan algoritma pembelajaran ansambel yang memadukan beberapa pohon keputusan untuk meningkatkan akurasi dan ketahanan model. Dalam konteks dataset kanker payudara, teknik Random Forest dapat digunakan untuk memprediksi apakah tumor tersebut ganas atau jinak, berdasarkan fitur masukan seperti ukuran, tekstur, dan bentuk tumor. Salah satu keuntungan menggunakan teknik Random Forest adalah kemampuannya dalam menangani sejumlah besar fitur masukan, serta mampu menghindari overfitting.

```
#RANDOM FOREST
from sklearn.inspection import permutation_importance
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier()
rf.fit(X_train, y_train)
scores = cross_val_score(rf, X_train, y_train, scoring='accuracy', cv=10).mean()
print("Accuracy %s" % round(scores*100,2))
result = permutation_importance(rf, X_train, y_train, n_repeats=10, random_state=42)
perm_sorted_idx = result.importances_mean.argsort()
tree_importance_sorted_idx = np.argsort(rf.feature_importances_)
tree_indices = np.arange(0, len(rf.feature_importances_)) + 0.5
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 8))
ax1.barh(tree_indices, rf.feature_importances_[tree_importance_sorted_idx], height=0.7)
ax1.set_yticks(tree_indices)
ax1.set_yticklabels(X_train.columns[tree_importance_sorted_idx])
ax1.set_ylim((0, len(rf.feature_importances_)))
ax2.boxplot(
    result.importances[perm_sorted_idx].T,
    vert=False,
    labels=X_train.columns[perm_sorted_idx],
)
fig.tight_layout()
plt.show()
```





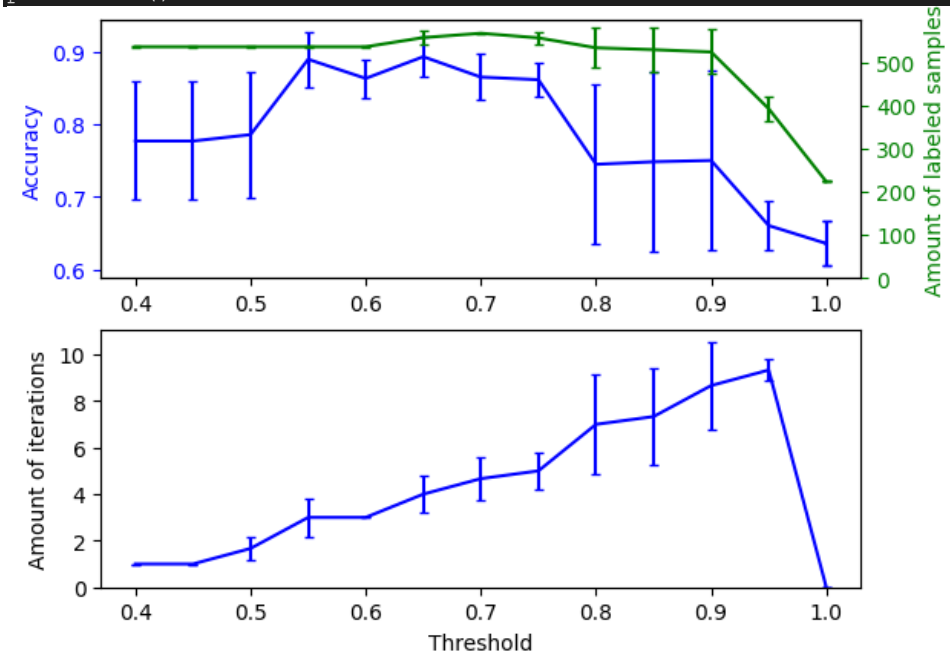
4. Self Training

Self Training merupakan algoritma pembelajaran semi-diawasi yang menggunakan data berlabel dan tidak berlabel untuk meningkatkan kinerja model. Dalam konteks kumpulan data kanker payudara, pelatihan mandiri dapat digunakan untuk mengklasifikasikan tumor sebagai ganas atau jinak berdasarkan fitur masukan. Algoritma ini pertama-tama melatih model pada data berlabel, lalu menggunakan model tersebut untuk memprediksi label dari data yang tidak berlabel. Selanjutnya, label yang diprediksi akan ditambahkan ke dalam data berlabel, dan model dilatih ulang pada data gabungan tersebut. Proses ini akan terus diulang hingga kinerja model konvergen.

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.svm import SVC
from sklearn.model_selection import StratifiedKFold
from sklearn.semi_supervised import SelfTrainingClassifier
from sklearn.metrics import accuracy_score
from sklearn.utils import shuffle

n_splits = 3

X, y = datasets.load_breast_cancer(return_X_y=True)
X, y = shuffle(X, y, random_state=42)
y_true = y.copy()
y[50:] = -1
total_samples = y.shape[0]

base_classifier = SVC(probability=True, gamma=0.001, random_state=42)
```

```
x_values = np.arange(0.4, 1.05, 0.05)
x_values = np.append(x_values, 0.99999)
scores = np.empty((x_values.shape[0], n_splits))
amount_labeled = np.empty((x_values.shape[0], n_splits))
amount_iterations = np.empty((x_values.shape[0], n_splits))

for i, threshold in enumerate(x_values):
    self_training_clf = SelfTrainingClassifier(base_classifier, thresho
ld=threshold)

    # We need manual cross validation so that we don't treat -
1 as a separate
    # class when computing accuracy
    skfolds = StratifiedKFold(n_splits=n_splits)
    for fold, (train_index, test_index) in enumerate(skfolds.split(X, y
)):
        X_train = X[train_index]
        y_train = y[train_index]
        X_test = X[test_index]
        y_test = y[test_index]
        y_test_true = y_true[test_index]

        self_training_clf.fit(X_train, y_train)

        # The amount of labeled samples that at the end of fitting
        amount_labeled[i, fold] = (
            total_samples
            - np.unique(self_training_clf.labeled_iter_, return_counts=
True)[1][0]
        )
        # The last iteration the classifier labeled a sample in
        amount_iterations[i, fold] = np.max(self_training_clf.labeled_i
ter_)

        y_pred = self_training_clf.predict(X_test)
        scores[i, fold] = accuracy_score(y_test_true, y_pred)


ax1 = plt.subplot(211)
ax1.errorbar(
    x_values, scores.mean(axis=1), yerr=scores.std(axis=1), capsize=2,
color="b"
)
ax1.set_ylabel("Accuracy", color="b")
ax1.tick_params("y", colors="b")

ax2 = ax1.twinx()
ax2.errorbar(
```

```
    x_values,
    amount_labeled.mean(axis=1),
    yerr=amount_labeled.std(axis=1),
    capsize=2,
    color="g",
)
ax2.set_ylim(bottom=0)
ax2.set_ylabel("Amount of labeled samples", color="g")
ax2.tick_params("y", colors="g")

ax3 = plt.subplot(212, sharex=ax1)
ax3.errorbar(
    x_values,
    amount_iterations.mean(axis=1),
    yerr=amount_iterations.std(axis=1),
    capsize=2,
    color="b",
)
ax3.set_ylim(bottom=0)
ax3.set_ylabel("Amount of iterations")
ax3.set_xlabel("Threshold")


plt.show()
```



5. Kesimpulan

Dalam laporan teknis ini, kami mengeksplorasi penggunaan Seaborn, Decision Tree, Random Forest, dan Self Training dalam analisis dataset kanker payudara. Seaborn digunakan untuk visualisasi data, sementara Decision Tree dan Random Forest digunakan untuk tugas klasifikasi. Selain itu, Self Training digunakan untuk meningkatkan kinerja model. Teknik-teknik ini dapat digunakan untuk menganalisis dataset kanker payudara dan memprediksi diagnosis tumor, yang dapat membantu dalam deteksi dini dan pengobatan kanker payudara.