SVM

October 5, 2020

Coding Assignment 6

```
SVM
Dataset: Pima Indians Diabetes Database
Author: Sreejith S
```

Dataset: https://www.kaggle.com/uciml/pima-indians-diabetes-database

```
[21]: import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

%reload_ext autoreload
%matplotlib inline
%autoreload 2
%config InlineBackend.figure_format = 'retina'


#classifiers
from sklearn.dummy import DummyClassifier
from sklearn import svm

from sklearn.model_selection import train_test_split

#evaluation metrics
from sklearn.metrics import plot_confusion_matrix, confusion_matrix,␣
 ↪ConfusionMatrixDisplay
from sklearn.metrics import plot_roc_curve, roc_curve, roc_auc_score
from sklearn.metrics import f1_score
from sklearn.metrics import plot_precision_recall_curve, precision_recall_curve,␣
 ↪precision_score, recall_score
from sklearn.metrics import matthews_corrcoef, average_precision_score


#plotting utils
from utils import plot_table, plot_confusion
```

[40]:

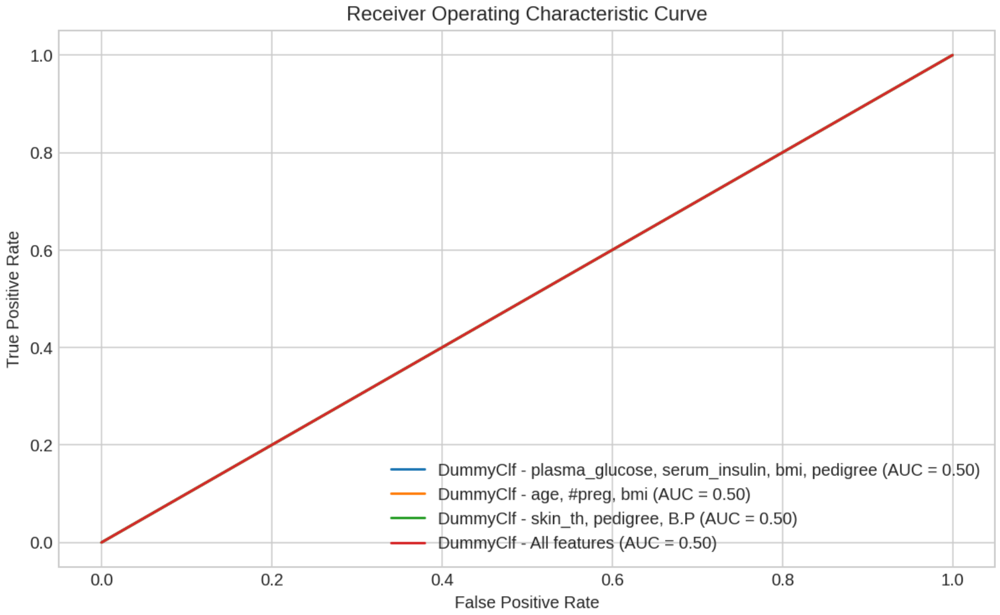
```
#set pd display options
pd.set_option('display.max_columns', 10)
pd.set_option('display.width', 80)

# Importing the dataset
dataset = pd.read_csv('../datasets/pima-indians-diabetes.csv')

#selecting features
f_best = ['plasma_glucose', 'serum_insulin', 'bmi', 'diab_pedigree']
f_a = ['age', 'pregnant', 'bmi']
f_b = ['skin_thickness', 'diab_pedigree', 'dia_BP']
f_all = ['pregnant', 'plasma_glucose', 'dia_BP', 'skin_thickness',␣
 ↪'serum_insulin', 'bmi', 'diab_pedigree', 'age']

def test_model(classifiers, feature_sets, short_titles, titles):
    """
    Arguments:
        classifiers - list of classifiers
        feature_sets - list of featuresets to be used
    """

    y = dataset[['Diab']].values.ravel()
    confusion_matrices = []
    eval_metrics_list = []


    fig = plt.figure(figsize=(10, 20))
    plt.style.use("seaborn-whitegrid")
    ax_roc = fig.add_subplot(3, 1, 1)
    ax_roc.set_title("Receiver Operating Characteristic Curve")
    ax_pre = fig.add_subplot(3, 1, 2)
    ax_pre.set_title("Precision-Recall Curve")

    ax_table = fig.add_subplot(3, 1, 3)
    ax_table.set_title("Evaluation Metrices")


    for i, (clf, feature) in enumerate(zip(classifiers, feature_sets)):
        X = dataset[feature]
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
                                                            random_state=0)
        clf.fit(X_train, y_train)
        y_preds = clf.predict(X_test)
        y_proba = clf.predict_proba(X_test)[:, 1]

        plot_roc_curve(clf, X_test, y_test, ax=ax_roc, name=titles[i])
        plot_precision_recall_curve(clf, X_test, y_test, ax=ax_pre,␣
↪name=titles[i])

        confusion_matrices.append(confusion_matrix(y_test, y_preds))
        eval_metrics = [clf.score(X_test, y_test),
                        precision_score(y_test, y_preds, zero_division=0),
                        recall_score(y_test, y_preds),
                        f1_score(y_test, y_preds),
                        matthews_corrcoef(y_test, y_preds),
                        roc_auc_score(y_test, y_proba),
                        average_precision_score(y_test, y_proba)]
        eval_metrics = [f"{i:.2f}" for i in eval_metrics]
        eval_metrics_list.append(eval_metrics)

    ax_pre.legend(loc='upper right')
    plot_table(eval_metrics_list, ax_table, short_titles)
    plot_confusion(confusion_matrices, short_titles)
    plt.show()
```

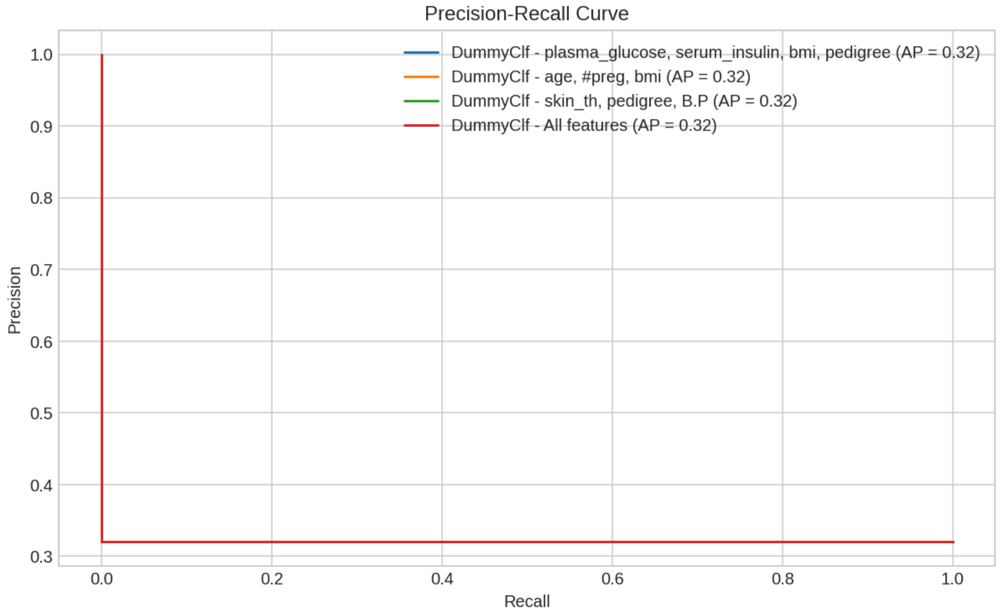
6.1. [1 marks] Create a dummy classifier (see model evaluation video from AnacondaCon), fit it using the three featuresets (a) all features, (b) Only age, pregnant, bmi and (c) only skin_thickness, diab_pedigree, and dia_BP as the input features.

Evaluate each of these above models using the metrics of precision, recall, F1-score, and AUROC (area under receiver operating characteristics).

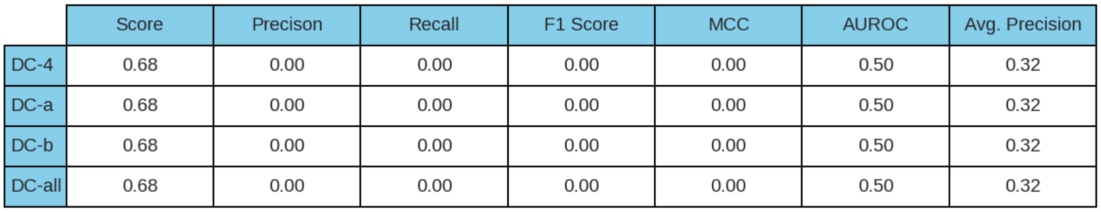
[41]:

```
short_titles = ["DC-4", "DC-a", "DC-b", "DC-all"]
titles = ["DummyClf - plasma_glucose, serum_insulin, bmi, pedigree",
          "DummyClf - age, #preg, bmi",
          "DummyClf - skin_th, pedigree, B.P",
          "DummyClf - All features"]
feature_sets = [f_best, f_a, f_b, f_all]

dummy =  DummyClassifier(strategy='most_frequent', random_state=0)
classifiers = [dummy, dummy, dummy, dummy]
test_model(classifiers, feature_sets, short_titles, titles)
```

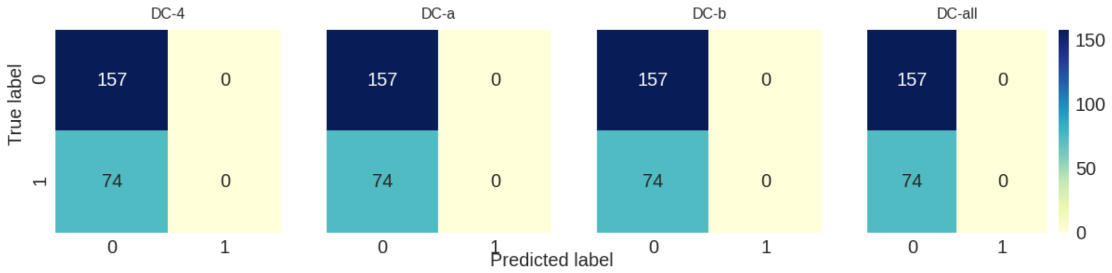




Evaluation Metrices



	Score	Precison	Recall	F1 Score	MCC	AUROC	Avg. Precision
DC-4	0.68	0.00	0.00	0.00	0.00	0.50	0.32
DC-a	0.68	0.00	0.00	0.00	0.00	0.50	0.32
DC-b	0.68	0.00	0.00	0.00	0.00	0.50	0.32
DC-all	0.68	0.00	0.00	0.00	0.00	0.50	0.32

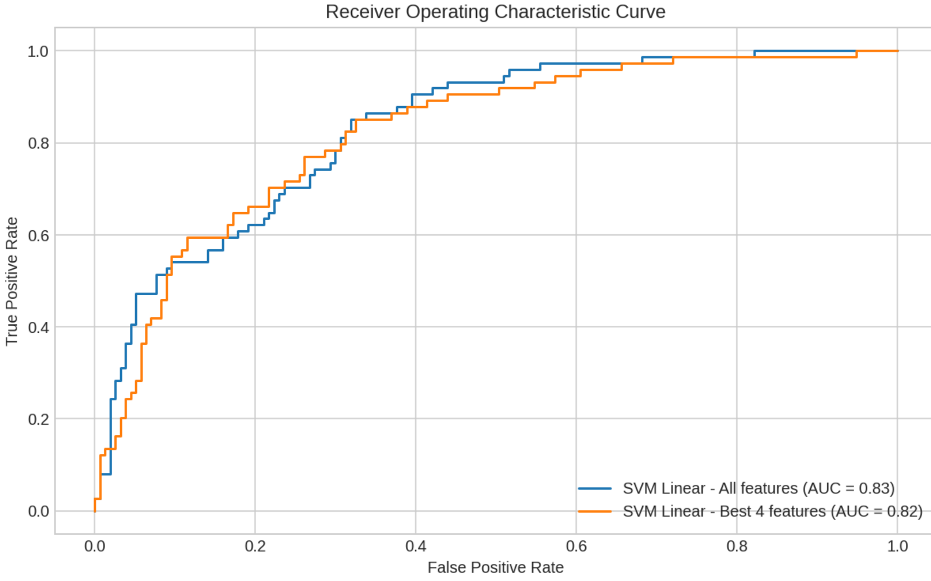


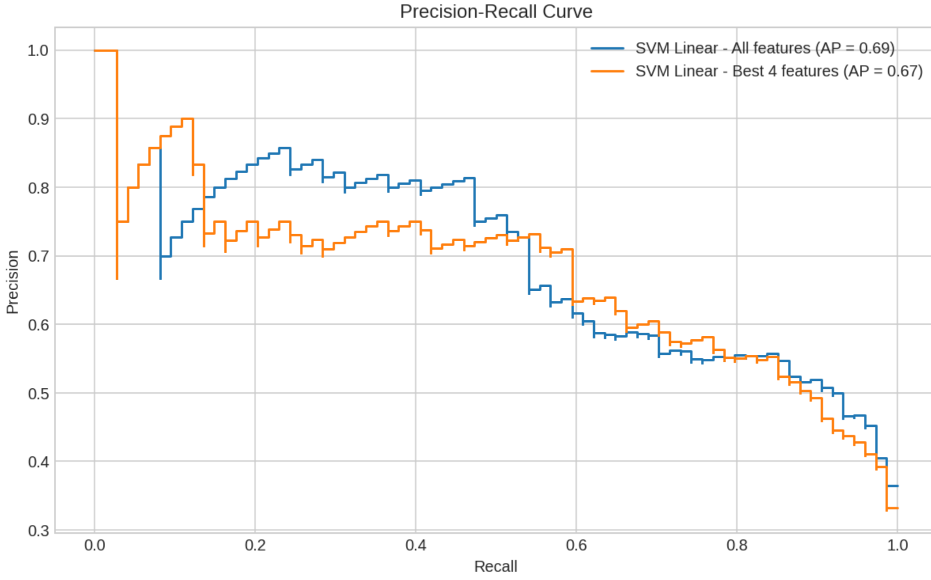
6.2. [3 marks] Write the code to build an SVM classification model with linear kernel. Use training data to fit and test data to evaluate. For this question, build the model using all features.

Evaluate the SVM model that you have created using the metrics of precision, recall, F1-score, and AUROC (area under receiver operating characteristics), and tabulate the results below.

[32]:

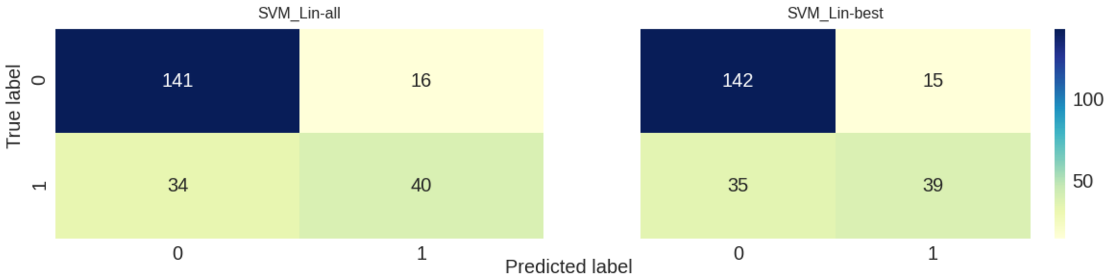
```
short_titles = ["SVM_Lin-all", "SVM_Lin-best"]
titles = ["SVM Linear - All features", "SVM Linear - Best 4 features"]
feature_sets = [f_all, f_best]

svm_clf = svm.SVC(kernel='linear', probability=True) #by default probability is disabled
classifiers = [svm_clf, svm_clf]
test_model(classifiers, feature_sets, short_titles, titles)
```





Evaluation Metrices

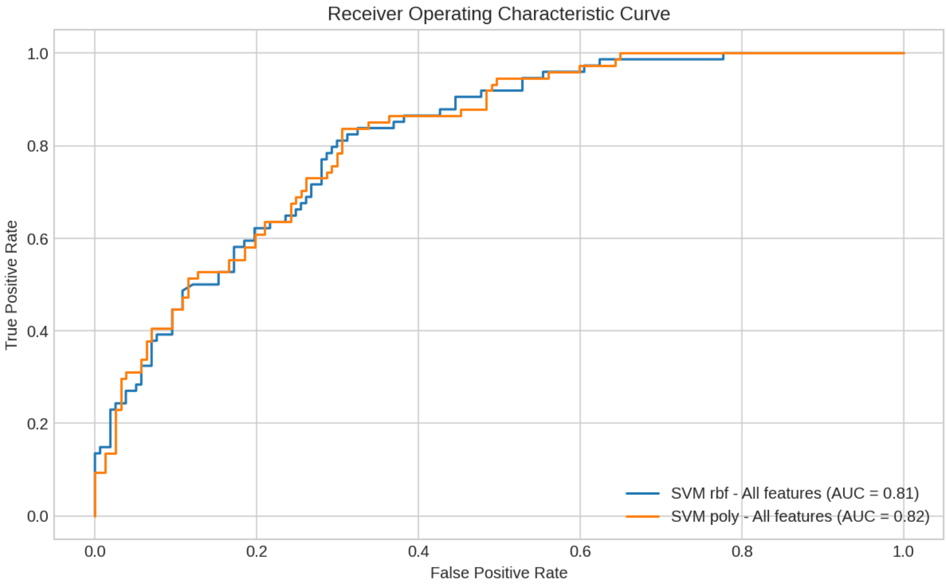
	Score	Precison	Recall	F1 Score	MCC	AUROC	Avg. Precision
SVM_Lin-all	0.78	0.71	0.54	0.62	0.48	0.83	0.69
SVM_Lin-best	0.78	0.72	0.53	0.61	0.48	0.82	0.67

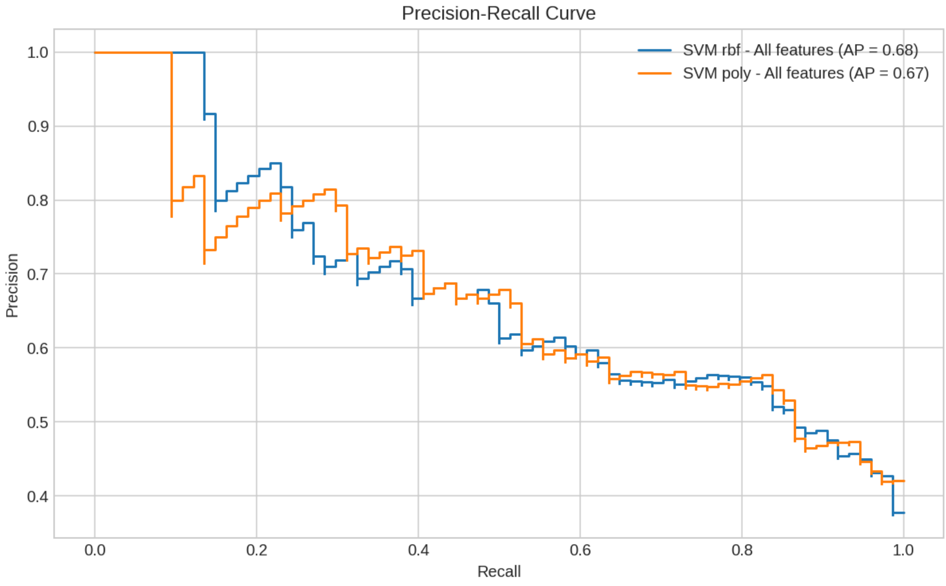


6.3. [2 marks] Write code to build SVM classification models with 'rbf' kernel and 'poly' kernel. Use training data to fit and test data to evaluate. For this question, build the model using all features. Evaluate the SVM model that you have created using the metrics of precision, recall, F1-score, and AUROC (area under receiver operating characteristics), and tabulate the results below.

[26]:

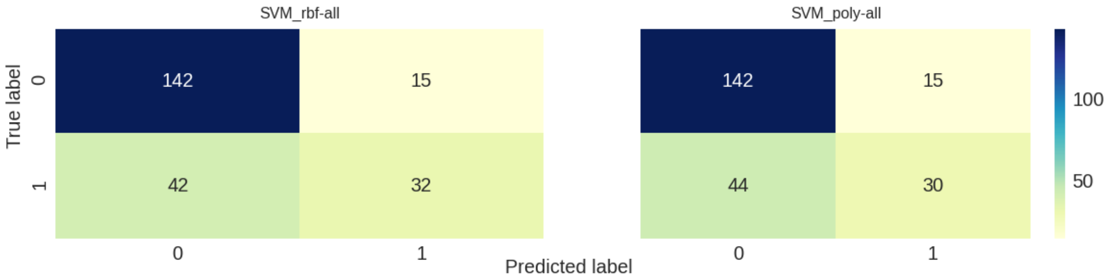
```
short_titles = ["SVM_rbf-all", "SVM_poly-all"]
titles = ["SVM rbf - All features", "SVM poly - All features"]
feature_sets = [f_all, f_all]

svm_clf_rbf = svm.SVC(kernel='rbf', probability=True)
svm_clf_poly = svm.SVC(kernel='poly', probability=True)

classifiers = [svm_clf_rbf, svm_clf_poly]
test_model(classifiers, feature_sets, short_titles, titles)
```





Evaluation Metrices

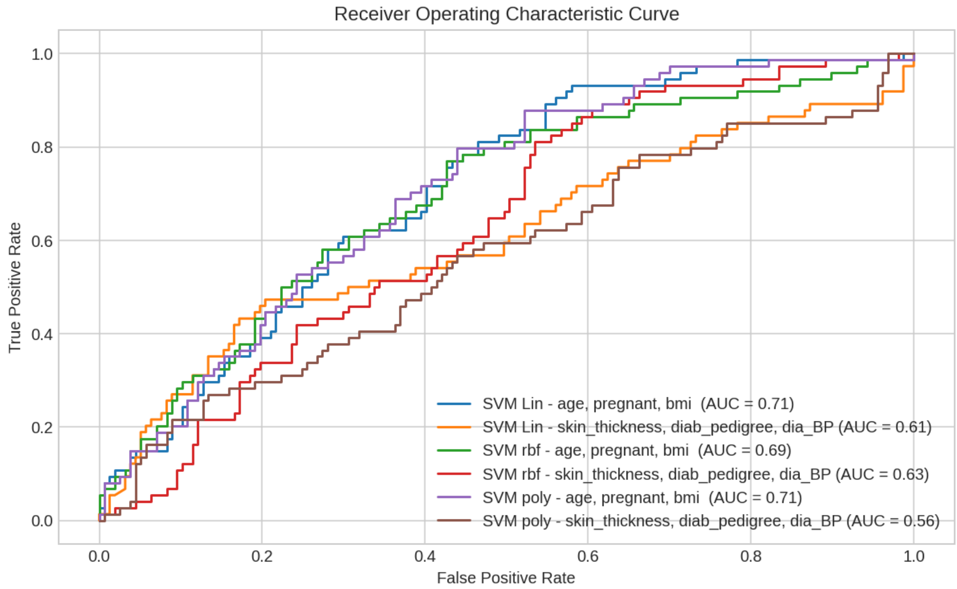
	Score	Precison	Recall	F1 Score	MCC	AUROC	Avg. Precision
SVM_rbf-all	0.75	0.68	0.43	0.53	0.39	0.81	0.68
SVM_poly-all	0.74	0.67	0.41	0.50	0.37	0.82	0.67

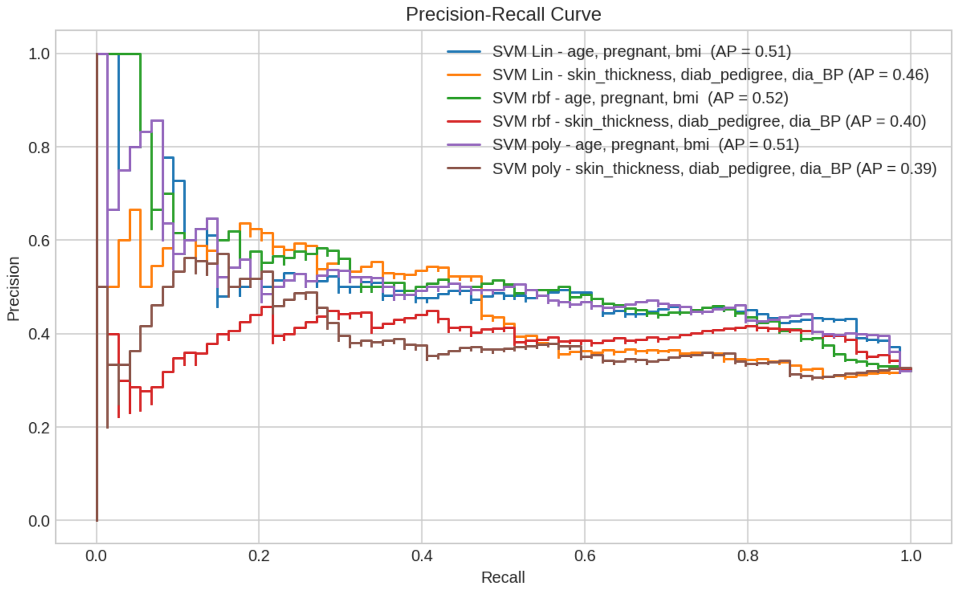


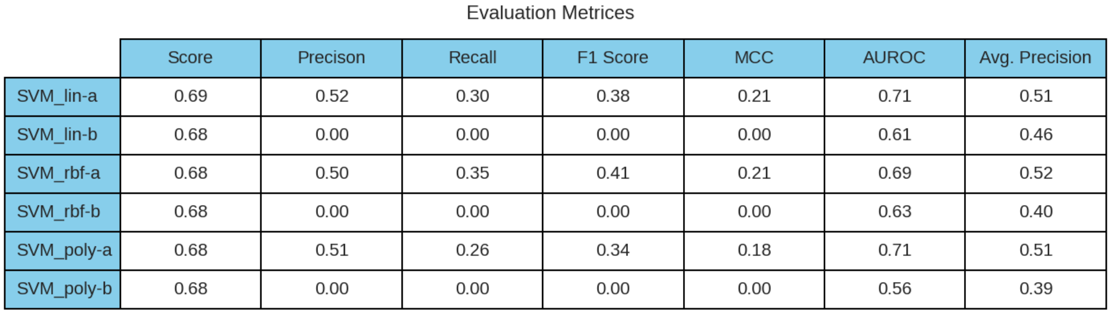
6.4. [3 marks] Write code to build SVM classification models with 'linear', 'rbf' and 'poly' kernels using input featuresets as: (a) Only age, pregnant, bmi and (b) only skin_thickness, diab_pedigree, and dia_BP. Use training data to fit and test data to evaluate. Evaluate the SVM model that you have created using the metrics of precision, recall, F1-score, and AUROC (area under receiver operating characteristics), and tabulate the results below.

[39]:

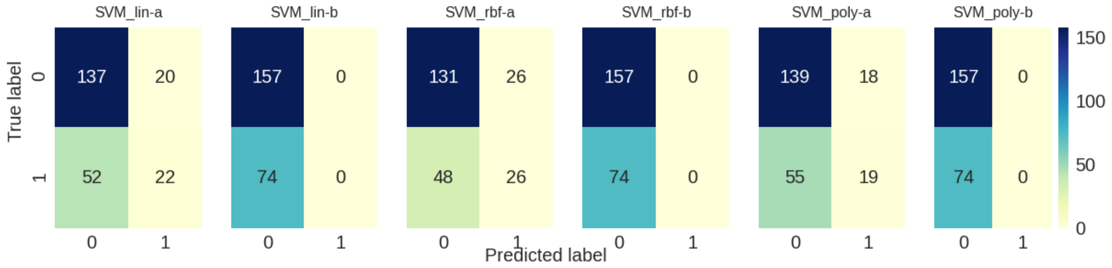
```
short_titles = ["SVM_lin-a", "SVM_lin-b", "SVM_rbf-a", "SVM_rbf-b",␣
 ↪"SVM_poly-a", "SVM_poly-b"]
titles = ["SVM Lin - age, pregnant, bmi ", "SVM Lin - skin_thickness,␣
 ↪diab_pedigree, dia_BP",
          "SVM rbf - age, pregnant, bmi ", "SVM rbf - skin_thickness,␣
 ↪diab_pedigree, dia_BP",
          "SVM poly - age, pregnant, bmi ", "SVM poly - skin_thickness,␣
 ↪diab_pedigree, dia_BP"]
feature_sets = [f_a, f_b,
                f_a, f_b,
                f_a, f_b]

svm_clf_lin = svm.SVC(kernel='linear', probability=True)
svm_clf_rbf = svm.SVC(kernel='rbf', probability=True)
svm_clf_poly = svm.SVC(kernel='poly', probability=True)

classifiers = [svm_clf_lin, svm_clf_lin, svm_clf_rbf, svm_clf_rbf, svm_clf_poly,␣
 ↪svm_clf_poly]
test_model(classifiers, feature_sets, short_titles, titles)
```


Receiver Operating Characteristic Curve


Precision-Recall Curve


Evaluation Metrices

	Score	Precison	Recall	F1 Score	MCC	AUROC	Avg. Precision
SVM_lin-a	0.69	0.52	0.30	0.38	0.21	0.71	0.51
SVM_lin-b	0.68	0.00	0.00	0.00	0.00	0.61	0.46
SVM_rbf-a	0.68	0.50	0.35	0.41	0.21	0.69	0.52
SVM_rbf-b	0.68	0.00	0.00	0.00	0.00	0.63	0.40
SVM_poly-a	0.68	0.51	0.26	0.34	0.18	0.71	0.51
SVM_poly-b	0.68	0.00	0.00	0.00	0.00	0.56	0.39



6.5. [1 mark] According to your analysis, answer the following with proper reasoning.

1. Which SVM classification model kernel performed the best in the classification task? `rbf` kernel performed marginally better than `linear` kernel when considering `recall` and `F1_score`, while `linear` has better precision.

2. Did any SVM model perform as bad as the dummy classifier? All kernels using the feature set `skin_thickness, diab_pedigree, dia_BP` performed similar to the `DummyClassifier` even though when looking at the `ROC` it becomes clear that the this feature set performs better on average across all threshold values than the `DummyClassifier`.

The reason for `precision=undefined` & `recall=0` when using features `skin_thickness, diab_pedigree, dia_BP` is most likely because $\theta^T f < 0$ and thus the predicitons are always $\hat{y} = 0$, which is also evident from the confusion matrices.

3. Which featureset (full, (a), (b)) performed the best in the classification? The full featureset performed the best.

[ ]: