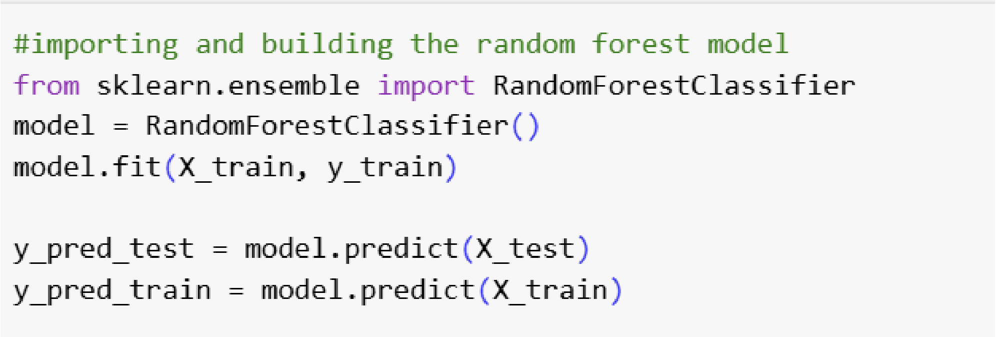
Model Development Phase Template

Date	18 June 2025
Team ID	SWTID1749880888
Project Title	Prosperity Prognosticator: Machine Learning for Startup Success Prediction
Maximum Marks	4 Marks

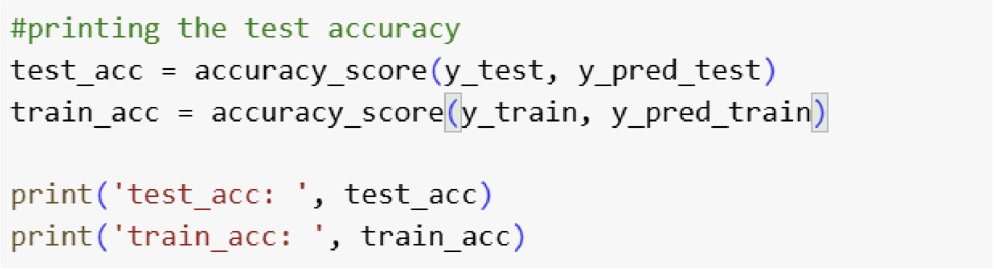
Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshots.

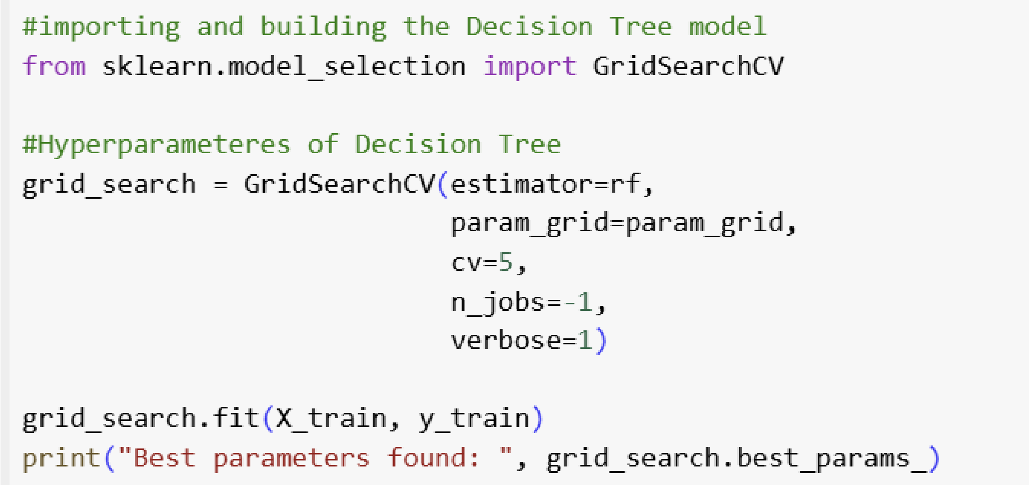
Initial Model Training Code:



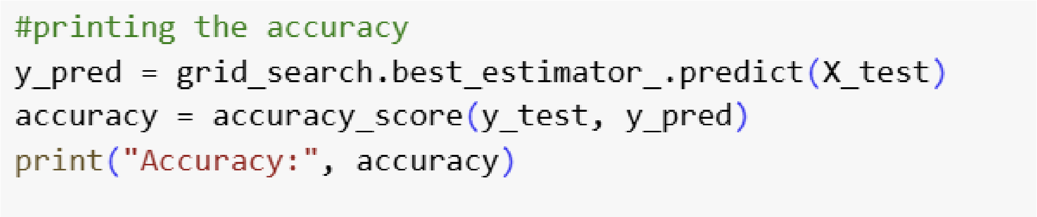
```
#importing and building the random forest model
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
model.fit(X_train, y_train)

y_pred_test = model.predict(X_test)
y_pred_train = model.predict(X_train)
```



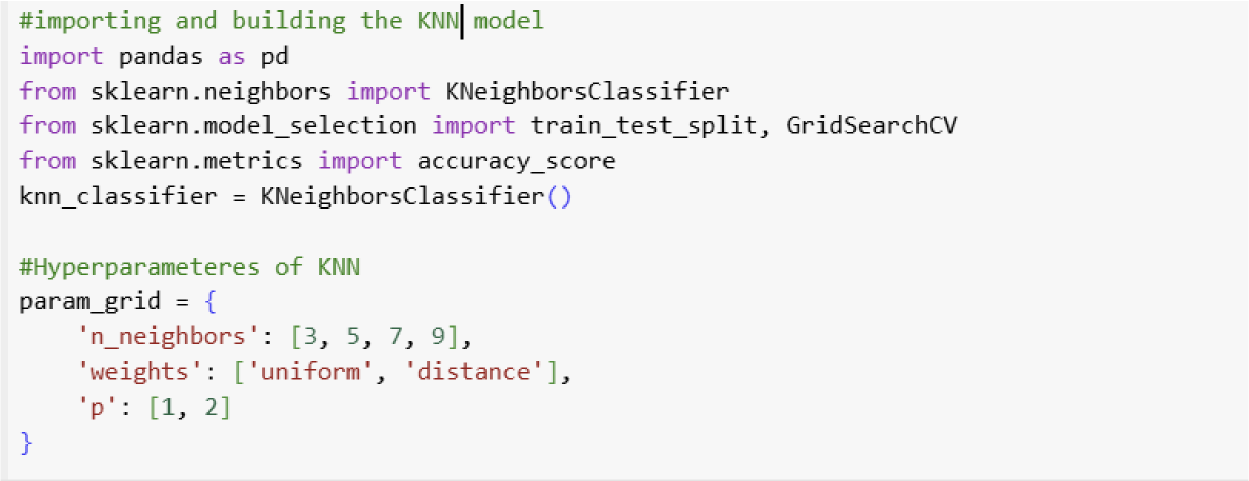
```
#printing the test accuracy
test_acc = accuracy_score(y_test, y_pred_test)
train_acc = accuracy_score(y_train, y_pred_train)

print('test_acc: ', test_acc)
print('train_acc: ', train_acc)
```



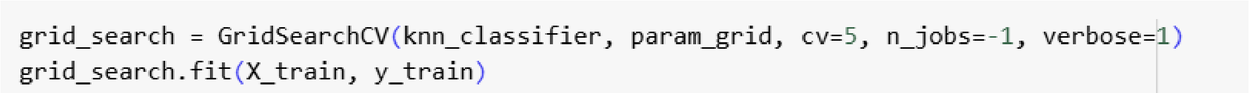
```
#importing and building the Decision Tree model
from sklearn.model_selection import GridSearchCV

#Hyperparameteres of Decision Tree
grid_search = GridSearchCV(estimator=rf,
                           param_grid=param_grid,
                           cv=5,
                           n_jobs=-1,
                           verbose=1)

grid_search.fit(X_train, y_train)
print("Best parameters found: ", grid_search.best_params_)
```



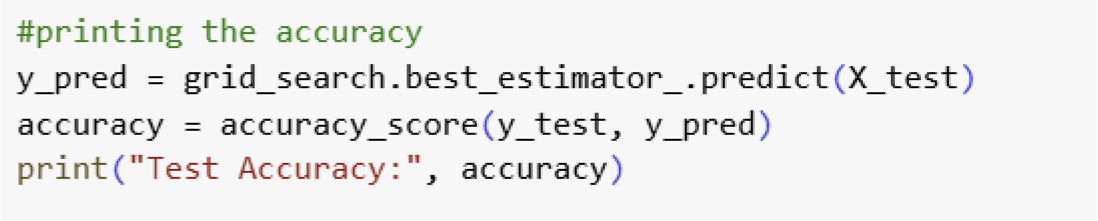
```
#printing the accuracy
y_pred = grid_search.best_estimator_.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```



```
#importing and building the KNN model
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import accuracy_score
knn_classifier = KNeighborsClassifier()

#Hyperparameteres of KNN
param_grid = {
    'n_neighbors': [3, 5, 7, 9],
    'weights': ['uniform', 'distance'],
    'p': [1, 2]
}
```

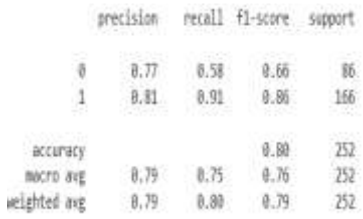
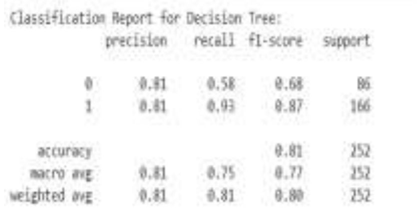
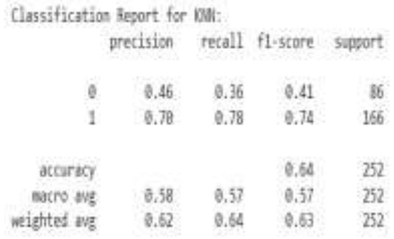


```
grid_search = GridSearchCV(knn_classifier, param_grid, cv=5, n_jobs=-1, verbose=1)
grid_search.fit(X_train, y_train)
```



```
#printing the accuracy
y_pred = grid_search.best_estimator_.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Test Accuracy:", accuracy)
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy	Confusion Matrix
Random forest	 precision recall f1-score support 0 0.77 0.58 0.66 86 1 0.81 0.91 0.86 166 accuracy 0.80 252 macro avg 0.79 0.75 0.76 252 weighted avg 0.79 0.80 0.79 252	**81%**	[[50 36] [15 151]]
Decision tree	 Classification Report for Decision Tree: precision recall f1-score support 0 0.81 0.58 0.68 86 1 0.81 0.93 0.87 166 accuracy 0.81 252 macro avg 0.81 0.75 0.77 252 weighted avg 0.81 0.81 0.80 252	**80%**	[[50 36] [12 154]]
Knn model	 Classification Report for KNN: precision recall f1-score support 0 0.46 0.36 0.41 86 1 0.70 0.78 0.74 166 accuracy 0.64 252 macro avg 0.58 0.57 0.57 252 weighted avg 0.62 0.64 0.63 252	**63%**	[[31 55] [36 130]]