Model Optimization and Tuning Phase Template

Date	July 2024
Team ID	739742
Project Title	Estimating the stock keeping units using Machine Learning
Maximum Marks	10 Marks

Model Optimization and Tuning Phase

The Model Optimization and Tuning Phase involves refining neural network models for peak performance. It includes optimized model code, fine-tuning hyperparameters, comparing performance metrics, and justifying the final model selection for enhanced predictive accuracy and efficiency.

Hyperparameter Tuning Documentation (8 Marks):

<table>
<tr><th>Model</th><th>Tuned Hyperparameters</th></tr>
<tr><td>Linear Regression</td><td>
#importing the library for grid search
from sklearn.model_selection import GridSearchCV

The ‘lr_param_grid’ specifies different values for regularization strength (C), solvers (solver), and penalty types (penalty). GridSearchCV (lr_cv) is employed with 5-fold cross-validation (cv=5), evaluating model performance based on accuracy (scoring="r2 score").

Linear Regression Hyperparameter Tunning

<code>from sklearn.model_selection import GridSearchCV</code>
<code>param_grid={'fit_intercept':[True,False],'copy_X':[True,False]}</code>
<code>grid_search=GridSearchCV(lr,param_grid,cv=5)</code>
<code>grid_search.fit(x_train,y_train)</code>

<code>GridSearchCV</code>
<code>estimator: LinearRegression</code>
<code>LinearRegression</code>

<code>pred_cv=grid_search.predict(x_test)</code>
</td></tr>
</table>

Random Forest

The parameter grid (make_regression) for hyperparameter tuning. It specifies different values for the number of trees (n_estimators), splitting criterion (criterion), maximum depth of trees (max_depth), and maximum number of features considered for splitting (max_features). GridSearchCV (rfc_cv) is employed with 3-fold cross-validation (cv=3), evaluating model performance based on accuracy (scoring="r2 score").

Random Forest Hyperparameter Tunning

```
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
from sklearn.datasets import make_regression
```

```
x,y=make_regression(n_samples=1000,n_features=10,random_state=42)
```

```
n_estimators=[int(x) for x in np.linspace(start=50,stop=250,num=10)]
max_features=['auto','sqrt']
max_depth=[int(x) for x in np.linspace(0,120,num=20)]
max_depth.append(None)
min_samples_split=[2,5,10]
min_samples_leaf=[1,2,4]
bootstrap=[True,False]
```

<table>
<tr>
<td>Decision

Tree</td>
<td>The parameters (params) define a randomized search for hyperparameter tuning of the Decision Tree Regressor (DecisionTreeRegressor), including max_depth, min_samples_leaf, min_samples_split and max_features . RandomizedSearchCV is used to evaluating model performance based on r2 score(scoring="r2 score")

Decision tree hyperparameter tunning

<pre>
from sklearn.model_selection import RandomizedSearchCV
param_dist={
    'max_depth':[None,5,10,15,20],
    'min_samples_split':[2,5,10],
    'min_samples_leaf':[1,2,4],
    'max_features':['auto','sqrt','log2']

}
tree=DecisionTreeRegressor()
dt=DecisionTreeRegressor()
dt_cv=RandomizedSearchCV(estimator=tree,param_distributions=param_dist
dt_cv.fit(x_train,y_train)
</pre></td>
</tr>
</table>

Final Model Selection Justification (2 Marks):

<table>
<tr><th>Final Model</th><th>Reasoning</th></tr>
<tr><td>Random Forest</td><td>
Random Forest model is chosen for its robustness in handling complex datasets and its ability to mitigate overfitting while providing high predictive r2 score.
<h2>Random Forest Regressor</h2>
<pre>from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error,r2_score</pre>
<pre>model=RandomForestRegressor()</pre>
<pre>model.fit(x_train,y_train)
pred=model.predict(x_test)</pre>
<pre>print("Mean Squared Error:",mean_squared_error(y_test,pr
print("R2 Score:",r2_score(y_test,pred))</pre>
<pre>Mean Squared Error: 892.5601685747586
R2 Score: 0.7279713962082139</pre>
Above all the models Random Forest model have the highest r2 score among all the models.
A higher r2 score is generally considered better as it indicates a more accurate and reliable model.
</td></tr>
</table>