Nội dung

1. Regression Analysis
2. Linenear regression
3. **Lựa chọn thuộc tính**

Lựa chọn thuộc tính

❑ Univariate Selection

```
# Univariate Selection
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import f_regression
```

```
#apply SelectKBest class to extract all best features
bestfeatures = SelectKBest(score_func=f_regression, k='all')
fit = bestfeatures.fit(inputs,outputs)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(inputs.columns)
```

```
#concat two dataframes for better visualization
featureScores = pd.concat([dfcolumns,dfscores],axis=1)
featureScores.columns = ['Specs','Score']  #naming the dataframe columns
print(featureScores.nlargest(3,'Score'))  #print 3 best features
```

```
         Specs        Score
2  petallength  1876.657813
0  sepallength   299.194957
1   sepalwidth    21.554378
```

https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html#sklearn.feature_selection.SelectKBest

Lựa chọn thuộc tính

```
# 2 features have highest scores
X_now = inputs[['petallength', 'sepallength']]
```

```
X_train_n, X_test_n, y_train_n, y_test_n = train_test_split(X_now, outputs, test_size=0.20)

regr_n = linear_model.LinearRegression()
regr_n = regr1.fit(X_train_n, y_train_n)
```

```
# The mean squared error
print("Mean squared error: %.2f"
      % mean_squared_error(outputs, regr_n.predict(X_now)))
# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % regr_n.score(X_now, outputs))
```

```
Mean squared error: 0.04
Variance score: 0.93
```

```
print("Train's score:", regr_n.score(X_train_n, y_train_n))
```

```
Train's score: 0.9363870023056706
```

```
print("Test's score:", regr_n.score(X_test_n, y_test_n))
```

```
Test's score: 0.8909006215218642
```

Lựa chọn thuộc tính

Feature Importance

```
from sklearn.ensemble import ExtraTreesRegressor
```

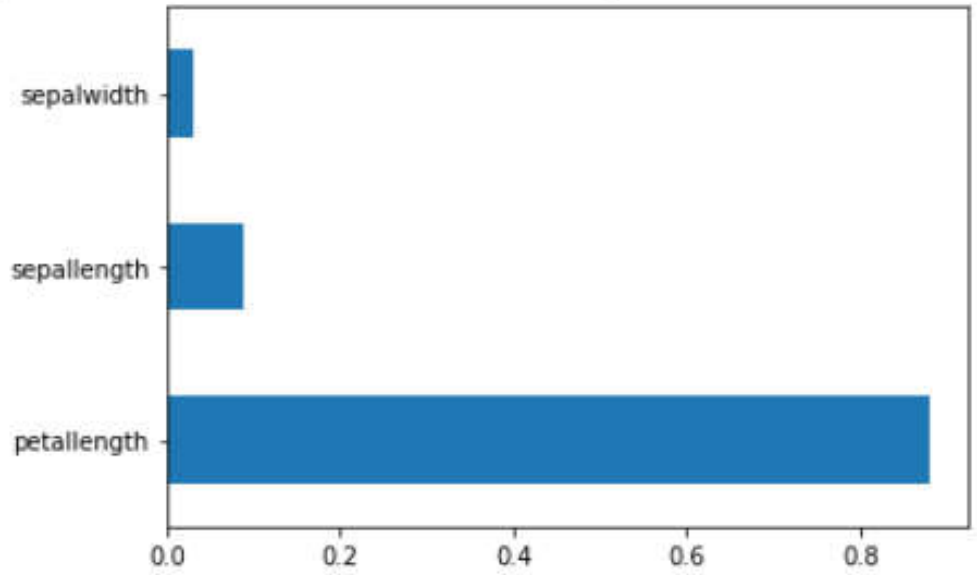
```
model = ExtraTreesRegressor()
model.fit(inputs,outputs)
```

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesRegressor.html

Lựa chọn thuộc tính

```
print(model.feature_importances_) #use inbuilt class feature_importances of tree based regressor
#plot graph of feature importances for better visualization
feat_importances = pd.Series(model.feature_importances_, index=inputs.columns)
feat_importances.nlargest(3).plot(kind='barh')
plt.show()
```

```
[0.08980434 0.03055877 0.87963689]
```



Lựa chọn thuộc tính

❑ Correlation Matrix with Heatmap

```
#get correlations of each features in dataset
data_sub = iris.iloc[:,0:4]
corrmat = data_sub.corr()
top_corr_features = corrmat.index
```

```
data_sub.corr()
```

	sepallength	sepalwidth	petallength	petalwidth
sepallength	1.000000	-0.109369	0.871754	0.817954
sepalwidth	-0.109369	1.000000	-0.420516	-0.356544
petallength	0.871754	-0.420516	1.000000	0.962757
petalwidth	0.817954	-0.356544	0.962757	1.000000

Lựa chọn thuộc tính

```
plt.figure(figsize=(10,10))
#plot heat map
g=sns.heatmap(data_sub[top_corr_features].corr(),cmap="RdYlGn", annot=True)
```