Team 3 - Data Science A

Machine Learning

Week #2

Kampus Merdeka x MyEduSolve

Feature Engineering

1. Feature Encoding
2. Feature Scaling
3. Feature Binning
4. Feature Selection

Feature Encoding

Machine learning models can only work with numerical values.

For this reason, it is necessary to transform the categorical values of the relevant features into numerical ones.

This process is called feature encoding.

Feature Encoding

There are several types of encoding that are often used

1. One-Hot Encoding
2. Ordinal Encoding
3. Label Encoding

One-Hot Encoding

One-hot encoding turns your categorical data into a binary vector representation.

We can do One-Hot Encoding on nominal data in several ways, but the easiest way is to use get_dummies()

One-Hot Encoding

datagy.io

Island		Biscoe	Dream	Torgensen
Biscoe	→	1	0	0
Torgensen		0	0	1
Dream		0	1	0

Ordinal Encoding

Ordinal encoding is a good choice if the order of the categorical variables matters.

For example, if we were predicting the price of a house, the label “small”, “medium”, and “large” would imply that a small house is cheaper than a medium house, which is cheaper than a large house.

The label is easily reversible and doesn’t increase the dimensionality of the data.

Original Encoding	Ordinal Encoding
Poor	1
Good	2
Very Good	3
Excellent	4

Label Encoding

Encode target labels with value between 0 and n_classes-1.

Label encoder is used when:

- The number of categories is quite large as one-hot encoding can lead to high memory consumption.
- When the order does not matter in categorical feature.

State (Nominal Scale)	State (Label Encoding)
Maharashtra	3
Tamil Nadu	4
Delhi	0
Karnataka	2
Gujarat	1
Uttar Pradesh	5

Feature Scaling

Feature Scaling is a technique to standardize the independent features present in the data in a fixed range.

If feature scaling is not done, then a machine learning algorithm tends to weigh greater values, higher and consider smaller values as the lower values, regardless of the unit of the values.

Feature Scaling

There are several types of scaling that are often used

1. Standard Scaler
2. MinMax Scaler
3. Robust Scaler

StandardScaler()

sklearn.preprocessing.StandardScaler

Standardize features by removing the mean and scaling to unit variance.

The standard score of a sample x is calculated as:
z = (x - u) / s

where u is the mean of the training samples or zero if with_mean=False, and s is the standard deviation of the training samples or one if with_std=False.

MinMaxScaler()

sklearn.preprocessing.MinMaxScaler

Transform features by scaling each feature to a given range.
This estimator scales and translates each feature individually such that it is in the given range on the training set, e.g. between zero and one.

The transformation is given by:
X_std = (X - X.min(axis=0)) / (X.max(axis=0) - X.min(axis=0))X_scaled = X_std * (max - min) + min

where min, max = feature_range.

This transformation is often used as an alternative to zero mean, unit variance scaling.

RobustScaler()

sklearn.preprocessing.RobustScaler

Scale features using statistics that are robust to outliers.

This Scaler removes the median and scales the data according to the quantile range (defaults to IQR: Interquartile Range).

The IQR is the range between the 1st quartile (25th quantile) and the 3rd quartile (75th quantile).

Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set. Median and interquartile range are then stored to be used on later data using the transform method.

Feature Binning

Binning or discretization is used for the transformation of a continuous or numerical variable into a categorical feature.

For example, if you have data about a group of people, you might want to organize their ages into smaller number of age intervals such as child, teens, adults, etc.

Feature Binning Example

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone	cut_age
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False	(20, 50]
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False	(20, 50]
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True	(20, 50]
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False	(20, 50]
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True	(20, 50]

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone	cut_age
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False	adult
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False	adult
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True	adult
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False	adult
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True	adult

Feature Selection

Feature Selection is the method of reducing the input variable to your model by using only relevant data and getting rid of noise in data.

It is the process of automatically choosing relevant features for your machine learning model based on the type of problem you are trying to solve.

Feature Selection

1. Filter Method
2. Embedded Method (Feature Importances)

Filter Method

The filter method evaluates each feature independently and then ranks the features after evaluating and picking the best.

The filter method uses statistical help to assign a score to each feature. Where each feature is ranked by score and selected to be kept or deleted from the dataset.

```
# contoh dengan ANOVA
bestfeatures = SelectKBest(score_func = f_classif, k=10) # jika regression gunakan f_regression
fit = bestfeatures.fit(x,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(x.columns)
```

```
# concat kedua dataframes
featureScores = pd.concat([dfcolumns, dfscores], axis = 1)
featureScores.columns = ['Specs', 'Score'] # memberi nama dataframe columns
```

```
#print hasil feature selection
featureScores.nlargest(10, 'Score') # print top 10 features
```

	Specs	Score
13	ram	3520.110824
0	battery_power	31.598158
12	px_width	22.620882
11	px_height	19.484842
8	mobile_wt	3.594318
6	int_memory	2.922996
9	n_cores	2.625415
14	sc_h	2.225984
15	sc_w	1.671000
16	talk_time	1.628811

Embedded Method (Feature Importances)

The embedded selector method is a feature selection method that combines the advantages of the filter method and the wrapper method.

Where the wrapper method requires one type of Machine Learning algorithm and uses performance as an evaluation criterion.

The method that is often used is the Tree Based model. We can use ExtraTreeClassifier for classification problems or ExtraTreeRegressor for regression problems.

```
# contoh embedded method dengan ExtraTree
model = ExtraTreesClassifier() # jika regression, gunakan ExtraTreeRegressor
model.fit(x,y) # fit model
print(model.feature_importances_) # gunakan inbuilt class: feature_importances
```

```
[0.06212889 0.01962479 0.03311334 0.01957469 0.03131533 0.01625055
 0.03387209 0.03129317 0.03504199 0.03233613 0.03305774 0.04738004
 0.04899331 0.40335455 0.03311449 0.03302993 0.03405608 0.01421776
 0.01837414 0.01987098]
```

```
# hasil feature importances
feat_importances = pd.Series(model.feature_importances_, index = x.columns)
feat_importances.nlargest(10)
```

```
ram              0.403355
battery_power    0.062129
px_width         0.048993
px_height        0.047380
mobile_wt        0.035042
talk_time        0.034056
int_memory       0.033872
sc_h             0.033114
clock_speed      0.033113
pc               0.033058
dtype: float64
```

Modeling

1. Linear Regression
2. Decision Tree

Simple Linear Regression

Simple Linear Regression is a type of Regression algorithms that models the relationship between a dependent variable and a single independent variable. The relationship shown by a Simple Linear Regression model is linear or a sloped straight line, hence it is called Simple Linear Regression.

The key point in Simple Linear Regression is that the dependent variable must be a continuous/real value. However, the independent variable can be measured on continuous or categorical values.

Simple Linear regression algorithm has mainly two objectives:

- Model the relationship between the two variables. Such as the relationship between Income and expenditure, experience and Salary, etc.
- Forecasting new observations. Such as Weather forecasting according to temperature, Revenue of a company according to the investments in a year, etc.

Multiple Linear Regression

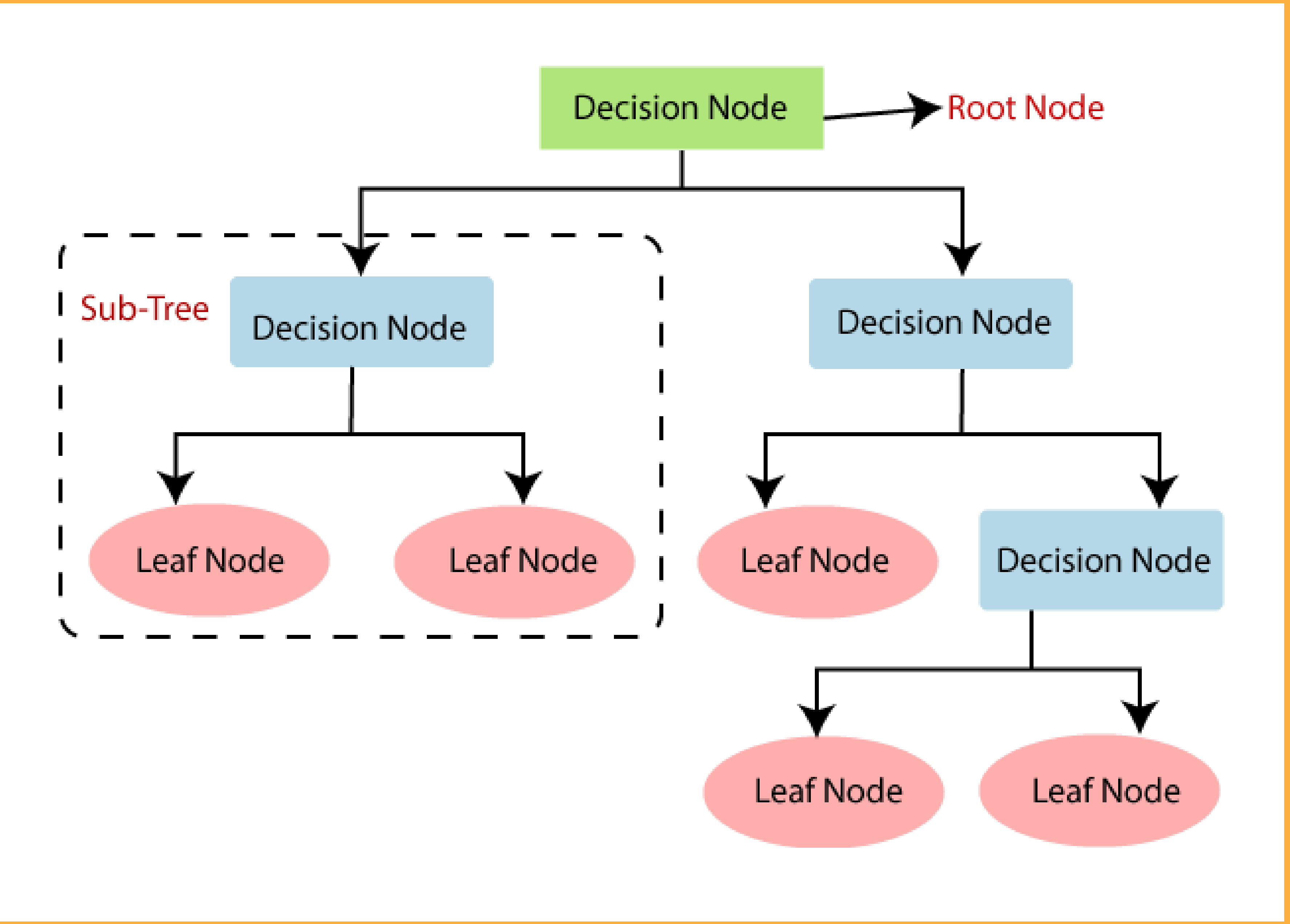
We can define it as:
"Multiple Linear Regression is one of the important regression algorithms which models the linear relationship between a single dependent continuous variable and more than one independent variable."

We have to do some classical assumption tests:
1. Linearity test, which is to test whether the variables X and y are linearly correlated. We can use the Pearson correlation test, Spearman, pairplot or scatterplot.
2. Normality Test (GOF), which is to test whether the variables are normally distributed (parametric). We can use test d'agostino(), Shapiro() or QQ-plot.
3. Multicollinearity, namely the existence of a strong correlation between the dependent variable (X). We can use variance_inflation_factor() or heatmap correlation.
4. Homoscedasticity, which is a condition where the variance of each residual value (error) is constant. We can use bartlett(), levene() or ANOVA tests.

Decision Tree

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems.

It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.



Thank you !