PRAKTIKUM MACHINE LEARNING

UNIT 6 : DATA RECONSTRUCTION

adult_klasifikasi.ipynb

Eka Praja Wiyata Mandala, S.Kom, M,Kom, CADS

Unit 6 : Data Reconstruction

Unit 6: Merekonstruksi Data

Tujuan: Mempersiapkan dan mengorganisir data untuk analisis dan pemodelan, termasuk feature engineering dan transformasi data.

Feature Engineering

Catatan:

ada potensi fitur engineering membuat korelasi tinggi.

Unit 6 : Data Reconstruction


```
# Membuat fitur baru berdasarkan usia
df['age_group'] = pd.cut(df['age'], bins=[0, 18, 30, 45, 60, 100], labels=
['Remaja', 'Dewasa Muda', 'Dewasa', 'Paruh Baya', 'Lansia'])

# Menggabungkan capital gain dan loss
df['net_capital'] = df['capital_gain'] - df['capital_loss']
```

```
# Membuat fitur rasio jam kerja terhadap rata-rata
df['work_hour_ratio'] = df['hours_per_week'] / df['hours_per_week'].mean()

# Membuat fitur kategorikal baru berdasarkan education_num
df['education_level'] = pd.cut(df['education_num'], bins=[0, 8, 12, 16, 20],
labels=['Dasar', 'Menengah', 'Sarjana', 'Pascasarjana'])

print(df[['age_group', 'net_capital', 'work_hour_ratio', 'education_level']].
head())
```

Unit 6 : Data Reconstruction


```
df.info()
```

```
df.head()
```

Unit 6 : Data Reconstruction

Transformasi Data

```
# Periksa kategori unik untuk kolom kategorikal
categorical_columns = df.select_dtypes(include=['object']).columns
for col in categorical_columns:
    print(f"\nKategori unik dalam {col}:")
    print(df[col].unique())
```

Unit 6 : Data Reconstruction


```python
# Lakukan Mapping data kategorikal menjadi numerik

# workclass
workclass = {'Never-worked':0, 'Without-pay':1, 'Self-emp-inc':2,
'Local-gov':3, 'Federal-gov':4, 'State-gov':5, 'Self-emp-not-inc':6,
'Private':7}

df['workclass'] = df['workclass'].map(workclass)

# education tidak diproses karena sudah dimapping
```

Unit 6 : Data Reconstruction


```
# maritalstatus
maritalstatus = {'Never-married':0, 'Married-civ-spouse':1, 'Divorced':2,
'Married-spouse-absent':3, 'Widowed':4, 'Married-AF-spouse':5, 'Separated':6}

df['marital_status'] = df['marital_status'].map(maritalstatus)

#occupation
occupation = {'Adm-clerical':1, 'Exec-managerial':2, 'Handlers-cleaners':3,
'Prof-specialty':4, 'Other-service':5, 'Sales':6, 'Craft-repair':7,
'Transport-moving':8, 'Farming-fishing':9, 'Machine-op-inspct':10,
'Tech-support':11, 'Protective-serv':12, 'Armed-Forces':13,
'Priv-house-serv':14}

df['occupation'] = df['occupation'].map(occupation)
```

Unit 6 : Data Reconstruction


```
#relationship
relationship = {'Unmarried':0, 'Not-in-family':1, 'Husband':2, 'Wife':3,
'Own-child':4, 'Other-relative':5}

df['relationship'] = df['relationship'].map(relationship)

#race
race ={'White':1, 'Black':2, 'Asian-Pac-Islander':3, 'Amer-Indian-Eskimo':4,
'Other':5}

df['race'] = df['race'].map(race)

# sex
sex = {'Female':0, 'Male':1}

df['sex'] = df['sex'].map(sex)
```

Unit 6 : Data Reconstruction


```
# nativecountry
nativecountry ={'United-States':1, 'Cuba':2, 'Jamaica':3, 'India':4,
'Mexico':5, 'South':6, 'Puerto-Rico':7, 'Honduras':8, 'England':9, 'Canada':10,
'Germany':11, 'Iran':12, 'Philippines':13, 'Italy':14, 'Poland':15,
'Columbia':16, 'Cambodia':17, 'Thailand':18, 'Ecuador':19, 'Laos':20,
'Taiwan':21, 'Haiti':22, 'Portugal':23, 'Dominican-Republic':24,
'El-Salvador':25, 'France':26, 'Guatemala':27, 'China':28, 'Japan':29,
'Yugoslavia':30, 'Peru':31, 'Outlying-US(Guam-USVI-etc)':32, 'Scotland':33,
'Trinadad&Tobago':34, 'Greece':35, 'Nicaragua':36, 'Vietnam':37, 'Hong':38,
'Ireland':39, 'Hungary':40, 'Holand-Netherlands':41}

df['native_country'] = df['native_country'].map(nativecountry)
```

```
# income
income = {'<=50K':0, '>50K':1}

df['income'] = df['income'].map(income)
```

Unit 6 : Data Reconstruction


```python
# Tambahan

# age_group
age_group = {'Remaja':1, 'Dewasa Muda':2, 'Dewasa':3, 'Paruh Baya':4,
'Lansia':5}
df['age_group'] = df['age_group'].map(age_group)

# education_level
education_level = {'Dasar':1, 'Menengah':2, 'Sarjana':3, 'Pascasarjana':4}
df['education_level'] = df['education_level'].map(education_level)
```

Unit 6 : Data Reconstruction


```python
df.head()
```

```python
# kita drop education karena sudah ada education_num
df = df.drop('education', axis=1)
```

```python
df
```

```python
df.info()
```

Unit 6 : Data Reconstruction

Rubah Menjadi Numerik

```
# df['nama_kolom'] = df['nama_kolom'].astype(str).astype(float)
df['age_group'] = df['age_group'].astype(str).astype(float)
df['education_level'] = df['education_level'].astype(str).astype(float)
```

```
df.info()
```

```
df.head()
```

Unit 6 : Data Reconstruction

Cek Kembali Korelasi

Karena semua fitur sudah bernilai numerik, maka bisa kita cek kembali korelasi semua fitur (termasuk target yang sebelumnya kategorikal)

```
# Correlation Heatmap
correlation = df.corr()
plt.subplots(figsize = (15,15))
sns.heatmap(correlation.round(2),
            annot = True,
            vmax = 1,
            square = True,
            cmap = 'RdYlGn_r')
plt.show()
```

Unit 6 : Data Reconstruction

Catatan:

- ada beberapa yang tidak ada nilainya (karena outlier detection)
- ada fitur berkorelasi tinggi (ada yang dari feature engineering)
- tidak ada korelasi tinggi terhadap target. (jika ada, maka harus dihapus. batasan = 0.9)
- untuk konteks klasifikasi, yang dicari adalah korelasi antar fitur rendah

Unit 6 : Data Reconstruction

Penghapusan Fitur yang Bernilai Konstan

```
df = df.loc[:,df.apply(pd.Series.nunique) != 1]
```

```
df
```

Unit 6 : Data Reconstruction

Penghapusan Fitur Berkorelasi Tinggi

```
# find and remove correlated features
def correlation(dataset, threshold):
    col_corr = set()  # Set of all the names of correlated columns
    corr_matrix = dataset.corr()
    for i in range(len(corr_matrix.columns)):
        for j in range(i):
            if abs(corr_matrix.iloc[i, j]) > threshold: # we are interested in
            absolute coeff value
                colname = corr_matrix.columns[i]  # getting the name of column
                col_corr.add(colname)
    return col_corr
```

Unit 6 : Data Reconstruction


```
data_tanpa_fitur = df.drop('income', axis=1)
```

```
corr_features = correlation(data_tanpa_fitur, 0.8)
print('correlated features: ', len(set(corr_features)) )
print(corr_features)
```

```
# removed correlated  features
df.drop(labels=corr_features, axis=1, inplace=True)
```

```
df.info()
```

```
df.describe()
```

Unit 6 : Data Reconstruction

Cek Kembali Korelasi

```
# Correlation Heatmap
correlation = df.corr()
plt.subplots(figsize = (15,15))
sns.heatmap(correlation.round(2),
            annot = True,
            vmax = 1,
            square = True,
            cmap = 'RdYlGn_r')
plt.show()
```