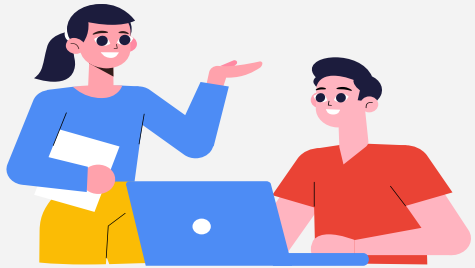
Machine Learning

STUDI INDEPENDEN

Kampus Merdeka
INDONESIA JAYA



Daftar Nama Kelompok:

- Setya Nugraha
- Rizki Maulana
- Mohammad Arif Mustofa

M T W T F

Contents

Stage 1

About Dataset

M T W T F

About Dataset

Dataset yang dimiliki berupa data set yang diambil dari sumber Kaggle.com yang berjudul “Classification Income” dimana ketika dataset dibaca dengan fungsi “pd.read_csv” akan tampil seperti pada gambar dibawah ini:

```
In [10]: pd.set_option('float_format', '{:.3f}'.format)

In [11]: df = pd.read_csv('income_evaluation.csv')
         df.head()
```

Out[11]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	income
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50K
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=50K
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	<=50K
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States	<=50K
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba	<=50K

Story Telling Dataset

Tujuan Project:

- Mengklasifikasikan pendapatan yang diperoleh setiap bulan
- Menentukan model algoritm yang terbaik

Tabel Dataset

Variabel	Tipe Data
Age	Int
Workclass	Obj
FNLWGT	Int
Eduation	Obj
Eduation – num	Int
Marital – status	Obj
Occupation	Obj
Relationship	Obj
Race	Obj
Sex	Obj
Capital – gain	Int
Capital – loss	Int
Hour – per – week	Int
Native – country	Obj
Income	Int

```
In [20]: # checking types of variables
         df.dtypes

Out[20]: age               int64
         workclass        object
         fnlwgt            int64
         education        object
         education-num     int64
         marital-status   object
         occupation       object
         relationship     object
         race             object
         sex              object
         capital-gain      int64
         capital-loss      int64
         hours-per-week    int64
         native-country   object
         income            int64
         dtype: object
```

Stage 3

Data Preprocessing

M T W T F

Exploratory Data

Cek Kolom Workclass apakah ada data yang missing

```
In [14]: df['workclass'].value_counts()

Out[14]: Private             22696
         Self-emp-not-inc     2541
         Local-gov            2093
         ?                    1836
         State-gov            1298
         Self-emp-inc         1116
         Federal-gov           960
         Without-pay            14
         Never-worked            7
         Name: workclass, dtype: int64
```

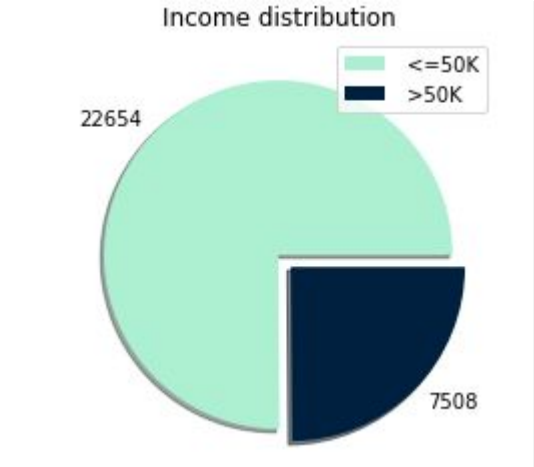
```
In [15]: shape0 = df.shape[0]
         for column in df.columns:
             df[column].replace(' ?', np.NaN, inplace=True)
         df = df.dropna().reset_index().drop(columns=['index'])
         shape1 = df.shape[0]
         print(str(shape0 - shape1) + ' rows have been removed.')

         2399 rows have been removed.

In [16]: df['workclass'].value_counts()

Out[16]: Private             22286
         Self-emp-not-inc     2499
         Local-gov            2067
         State-gov            1279
         Self-emp-inc         1074
         Federal-gov           943
         Without-pay            14
         Name: workclass, dtype: int64
```

Data yang didapatkan ternyata pada kolom Workclass terdapat missing berupa “?” sehingga dilakukan removed sebanyak 2399 baris data.

Data Visualization

Data visualisasi dengan diagram lingkaran

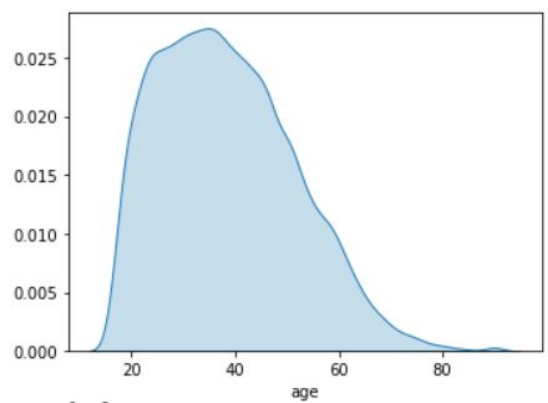
```
In [18]: colors = ['#ADEFD1FF', '#00203FFF']
         explode = [0, 0.1]
         plt.pie(income, labels=income.values, colors=colors, explode = explode, shadow=True)
         plt.title('Income distribution')
         plt.legend(labels=income.index)

Out[18]: <matplotlib.legend.Legend at 0x2c24294d280>
```



Didapatkan hasil untuk klasifikasi jumlah masyarakat yang memiliki pendapatan diatas 50k sebanyak 7508 dan dibawah 50k sebanyak 22654 orang.

Data Visualization

```
# membuat grafik
colors = list(sns.color_palette("Paired"))
fig, ax = plt.subplots(nrows=6, ncols=2, figsize=(12,28))
for i in range(6):
    sns.kdeplot(stats.iloc[:, i],
                shade = True,
                color = colors[i*2+1],
                ax=ax[i, 0]).set(ylabel = '', xlabel = stats.columns[i])
plt.show()
```

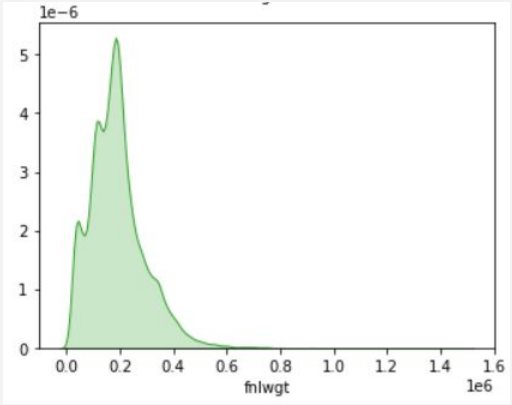


(a)

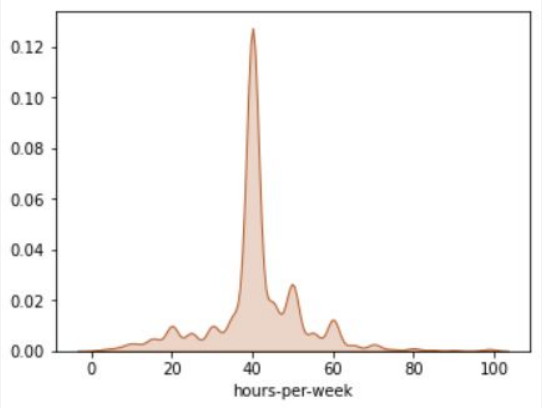
Selanjutnya dibuat grafik untuk melihat perbandingan antara variable data terhadap klasifikasi yang memilki pendapatan diatas 50.000.

(a) Nilai pendapatan diatas USD 50.000 terhadap variable umur

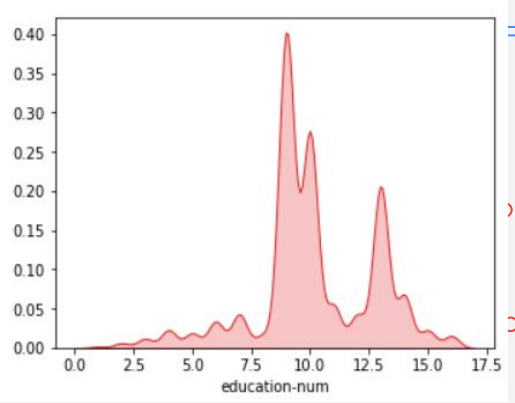
Data Visualization



(b)



(c)



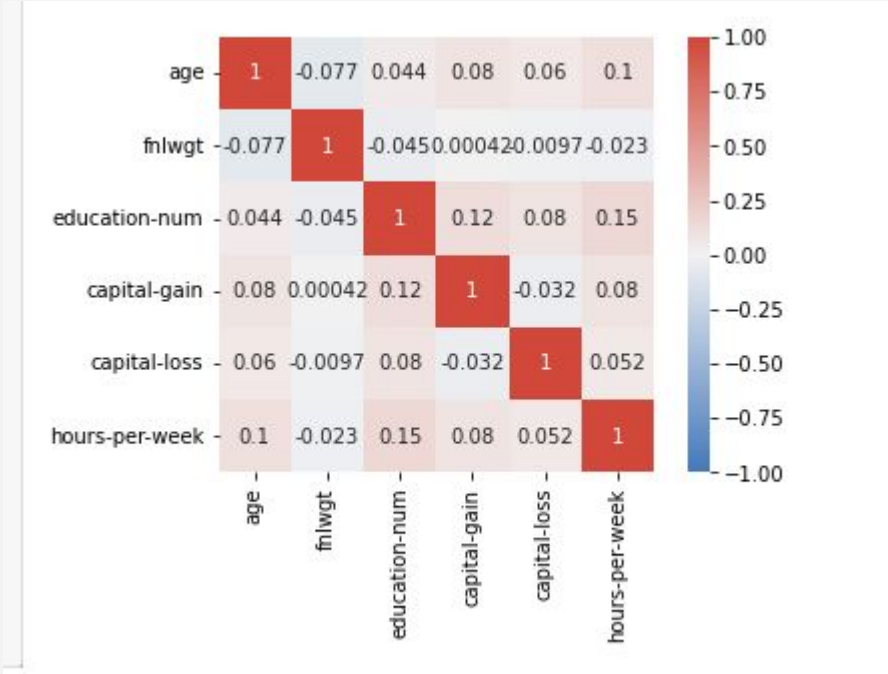
(d)

(b) Nilai pendapatan diatas USD 50.000 terhadap variable fnlwgt
(c) Nilai pendapatan diatas USD 50.000 terhadap variable hour-per-week (banyaknya jam kerja selama 1 pekan)
(d) Nilai pendapatan diatas USD 50.000 terhadap variable tingkatan pendidikan

Feature Engineering & Feature Selection

Correlation Heatmap

Menentukan hubungan kedekatan data (korelasi) dengan data yang lainnya. Ini dilakukan untuk menentukan mana fitur yang sebaiknya dihilangkan agar load data lebih akurat dan memiliki performa terbaik.



Normalisasi Data

Metode normalisasi data yang digunakan adalah Min-Max Scalar. Cara kerjanya setiap nilai pada sebuah fitur dikurangi dengan nilai minimum fitur tersebut, kemudian dibagi dengan rentang nilai atau nilai maksimum dikurangi nilai minimum dari fitur tersebut.

```
# creating scaler and new standardized train and test data frames.
scaler = MinMaxScaler()

X_train_numeral = scaler.fit_transform(X_train.select_dtypes(['float', 'int64']))
X_train_s = pd.concat([pd.DataFrame(X_train_numeral, index=X_train.index, columns=X_train.columns[:6]),
                       X_train.iloc[:, 6:]], axis=1)

X_test_numeral = scaler.transform(X_test.select_dtypes(['float', 'int64']))
X_test_s = pd.concat([pd.DataFrame(X_test_numeral, index=X_test.index, columns=X_test.columns[:6]),
                      X_test.iloc[:, 6:]], axis=1)
```

Stage 5

Modelling And Evaluation

M T W T F

Train Test Split

```
X = df_final.drop(columns=['income'])
y = df_final['income']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
```

```
# membuat scaler dan standardisasi baru untuk dataframe train dan test.
scaler = MinMaxScaler()

X_train_numeral = scaler.fit_transform(X_train.select_dtypes(['float', 'int64']))
X_train_s = pd.concat([pd.DataFrame(X_train_numeral, index=X_train.index, columns=X_train.columns[:6]),
                       X_train.iloc[:, 6:]], axis=1)

X_test_numeral = scaler.transform(X_test.select_dtypes(['float', 'int64']))
X_test_s = pd.concat([pd.DataFrame(X_test_numeral, index=X_test.index, columns=X_test.columns[:6]),
                      X_test.iloc[:, 6:]], axis=1)
```

```
X_train.head()
```

Split data dilakukan dengan membagi data menjadi
Data Trainning dan Data Testing

Fitting Data

```
# membuat fungsi yang digunakan untuk menyesuaikan dan memprediksi algoritme yang diberikan melalui list.
def algoritm_score_list(show_processing=False, standardized=False):
    scores_list = []

    for algorithm in algorithms:
        if show_processing:
            print('processing ' + str(algorithm) + ' algorithm...')

        if standardized:
            X_tn = X_train_s
            X_tt = X_test_s
        else:
            X_tn = X_train
            X_tt = X_test

        A = algorithm.fit(X_tn, y_train)
        y_predict = A.predict(X_tt)
        accuracy = accuracy_score(y_test,y_predict)

        scores_list.append([A, accuracy, standardized])

    print('all predictions finished')
    return scores_list
```

Evaluasi Model Klasifikasi

Model Algoritm	Accuracy	Accuracy Standard
Decision Tree Classifier	0.814	0.814
K-Near Neighbors	0.769	0.834
Logistic Regression	0.792	0.851

Seperti yang terlihat setelah data dilakukan standardisasi terjadi peningkatan akurasi pada K-Nneighbors dan Logistic Regression yang sesuai dangan teori. Sedangkan pada model Decision Tree data yang standar maupun tidak standar tidak mempengaruhi akurasi.
Dalam contoh berikut data standar akan digunakan jika ada akurasi yang lebih besar setelah standarisasi, sehingga akan dilakukan pemodelan menggunakan Logistic Regression dan KNNeighbors.

Kesimpulan

Dari pengujian akurasi dataset oleh masing – masing algoritma tersebut dapat disajikan pada tabel. Berdasarkan nilai *accuracy* algoritma yang lebih akurat adalah DecisionTree dan LogisticRegression disusul oleh k-Nearest Neighbor.

Selain itu, Algoritma DecisionTree dan Logistic Regression tidak memerlukan proses data yang di standarisasi terlebih dahulu agar dapat bekerja dengan baik. Sedangkan untuk KNearestNeighbors, dengan dilakukan proses standarisasi data meningkatkan akurasi sebesar 4,6%.

Sekian

Kampus Merdeka
INDONESIA JAYA

M
T
W
T
F