CRISP-DM

(Cross-Industry Standard Process for Data Mining)

Mechine Learning
Supervise Classification

By : Muhammad Hasbi Asyiddiki

https://github.com/asyiddiki

https://www.linkedin.com/in/hasbi3108

Outline Scope list

- **Business Understanding**
- ***Data Understanding***
- ***Data Preparation***
- ***ML Modeling***
- ***Evaluasi***
- ***Deployment***

Business Understanding

Company R adalah Agen penempatan kerja yang menyediakan program pelatihan data science gratis ke berbagai industri company client di seluruh Indonesia. Adapun persyaratan kandidat yang diberlakukan adalah sebagai berikut : (Mengidentifikasi orang-orang yang ingin pindah role menjadi target company R)

- Terbatas untuk 100 pelamar.
- Diselenggarakan secara triwulanan (dimulai pada minggu pertama bulan Januari, April, Juli, Okt).
- Durasi pelatihan 3 bulan.
- Agensi menawarkan kontrak penempatan kerja selama satu tahun.
- Company R mendapatkan keuntungan sebesar 0,25% dari gaji bulanan.
- Pekerjaan akan berakhir setelah periode satu tahun selesai, kecuali jika ditawarkan kontrak/perpanjangan baru

Membangun **model Machine Learning** yang dapat membantu mengidentifikasi apakah kandidat yang mendaftar pelatihan sedang mencari **perubahan pekerjaan atau tidak** berdasarkan dari data baru yang diberikan. Kemampuan untuk menyaring kandidat yang mendaftar pelatihan untuk mencari perubahan pekerjaan dengan cara yang lebih tepat. Porsi yang lebih besar dari peserta pelatihan yang mencari perubahan pekerjaan = Lebih banyak kandidat untuk program penempatan kerja

Goal :

Membantu Company R dalam meningkatkan keuntungan yang diperoleh dari **program penempatan kerja** dan memastikan biaya digunakan untuk **keuntungan yang lebih besar**.

Business Mactics :
Revenue (pendapatan) dari program penempatan kerja.

Data Understanding

Mengumpulkan data-data feature yang relevan dengan output yang di harapkan, pada kondisi ini ada 13 feature dan 1 label yang akan di gunakan. Kita memiliki 19158 data yang akan digunakan yang masing - masing memiliki 14 kolom. Kolom - kolom tersebut memiliki arti :

1. **enrollee_id** : ID unik kandidat
2. **City** : kota
3. **city_development _index** : Indeks perkembangan kota (berskala)
4. **Gender** : Jenis kelamin kandidat
5. **relevent_experience** : Pengalaman kandidat yang relevan
6. **enrolled_university** : Jenis program Universitas yang didaftarkan jika ada
7. **education_level** : Tingkat pendidikan kandidat
8. **major_discipline** : Disiplin ilmu kandidat / jurusan
9. **experience** : Pengalaman kandidat dalam beberapa tahun
10. **company_size** : Jumlah karyawan pada perusahaan pemberi kerja
11. **company_type** : Jenis perusahaan pemberi kerja
12. **last_new_job** : Selisih tahun antara pekerjaan sebelumnya dan pekerjaan saat ini
13. **training_hours** : lama waktu pelatihan selesai
14. **target**: 0 – Tidak mencari perubahan pekerjaan (not looking for job), 1 – Mencari perubahan pekerjaan (looking for job)

[11]:
```
# Load dataset
df = pd.read_csv(nama_file)

# Melihat tampilan dataset
df.head() # df.head(10)
```

[11]:

	enrollee_id	city	city_development_index	gender	relevent_experience	enrolled_university	education_level	major_discipline	experience	company_size	company_type	last_new_job
0	8949	city_103	0.920	Male	Has relevent experience	no_enrollment	Graduate	STEM	>20	NaN	NaN	1
1	29725	city_40	0.776	Male	No relevent experience	no_enrollment	Graduate	STEM	15	50-99	Pvt Ltd	>4
2	11561	city_21	0.624	NaN	No relevent experience	Full time course	Graduate	STEM	5	NaN	NaN	never
3	33241	city_115	0.789	NaN	No relevent experience	NaN	Graduate	Business Degree	<1	NaN	Pvt Ltd	never
4	666	city_162	0.767	Male	Has relevent experience	no_enrollment	Masters	STEM	>20	50-99	Funded Startup	4

Data Preparation

Label pada data yang kita miliki pada kolom target untuk mengidentifikasi seorang kandidat Tidak mencari perubahan pekerjaan (not looking for job), dan 1 – Mencari perubahan pekerjaan (looking for job)

Pada data yang kita miliki, maka di bagi mana sajakah dat yang bersifat numerik dan kategori agar kita bisa melakukan analisa leboh komprehensif

[42]:
```
# Membagi kolom menjadi kolom numerik dan kategori
var_kategori = [var for var in df.columns if df[var].dtype=='O' and var!=target]
var_numerik = [var for var in df.columns if df[var].dtype!='O' and var!=target]

# Melihat variabel yang sudah dipisah
print(var_kategori)
print(var_numerik)
```

```
['city', 'gender', 'relevent_experience', 'enrolled_university', 'education_level', 'major_discipline', 'experience', 'company_size', 'company_type', 'last_new_job']
['city_development_index', 'training_hours']
```

```
[17]: # Kita lihat ringkasan data
      df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19158 entries, 0 to 19157
Data columns (total 13 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   city                    19158 non-null  object
 1   city_development_index  19158 non-null  float64
 2   gender                  14650 non-null  object
 3   relevent_experience     19158 non-null  object
 4   enrolled_university     18772 non-null  object
 5   education_level         18698 non-null  object
 6   major_discipline        16345 non-null  object
 7   experience              19093 non-null  object
 8   company_size            13220 non-null  object
 9   company_type            13018 non-null  object
 10  last_new_job            18735 non-null  object
 11  training_hours          19158 non-null  int64
 12  target                  19158 non-null  float64
dtypes: float64(2), int64(1), object(10)
memory usage: 1.9+ MB
```

Setelah itu makukan preprosesing data dengan cara Dropna data yang memiliki nilai <0,5 % dari jumlah data. Pada kondisi ini ada 4 data yang memiliki nilai Null < 0,5 dari jumlah data. Jumlah data awal 19158 --> 18014

[28]:
```
# Menyeleksi hanya yang memiliki proporsi < 0.05 dan menyimpannya sebagai list var_cca
var_cca = [var for var in var_kosong if df[var].isnull().mean() < 0.05]
var_cca
```

[28]:
```
['enrolled_university', 'education_level', 'experience', 'last_new_job']
```

[29]:
```
# Menghapus data dengan method dropna() dan menambahkan parameter subset
# Parameter subset diisi dengan list nama kolom yang ingin dihapus nilai kosongnya
# Kita simpan sebagai tabel baru dengan nama df_cca
df_cca = df.dropna(subset=var_cca)

# Jika ingin menghapus semua baris data yang 'missing' tanpa memperdulikan variabelnya
# maka lakukan cara berikut
# df = df.dropna()

# Membandingkan dimensi df (baris, kolom) sebelum dan sesudah CCA
df.shape, df_cca.shape
```

[29]:
```
((19158, 13), (18014, 13))
```

[22]:
```
# Melihat total data kosong/baris dan berapa persen %
informasi = pd.DataFrame({'jumlah': jumlah, 'Percent': rata2})
informasi
```

[22]:

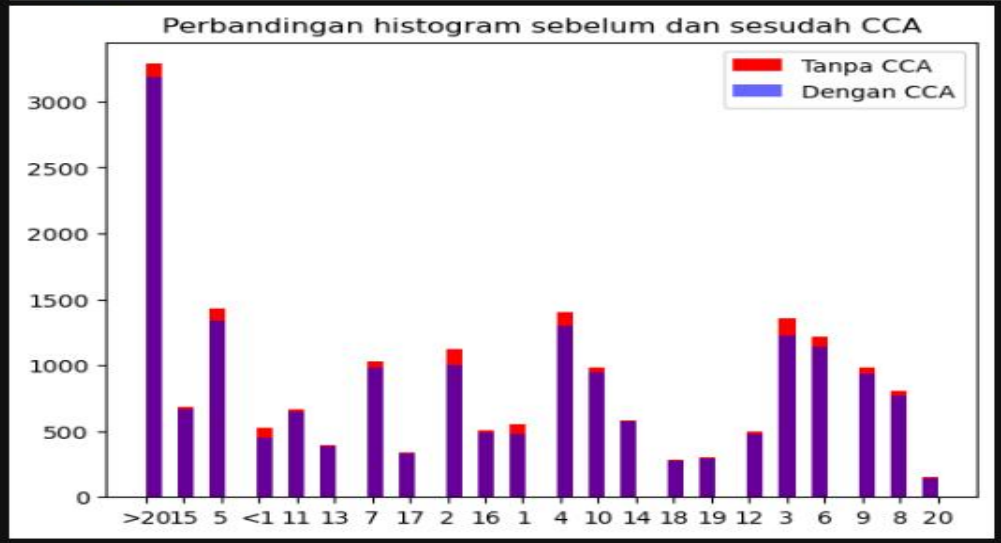
	jumlah	Percent
city	0	0.000000
city_development_index	0	0.000000
gender	4508	0.235306
relevent_experience	0	0.000000
enrolled_university	386	0.020148
education_level	460	0.024011
major_discipline	2813	0.146832
experience	65	0.003393
company_size	5938	0.309949
company_type	6140	0.320493
last_new_job	423	0.022080
training_hours	0	0.000000
target	0	0.000000

[34]:
```
# Jika ingin menghapus semua baris data yang 'm
# maka lakukan cara berikut
#df = df.dropna(axis=0) # HATI-HATI, AKAN MEMBU
```

[35]:
```
# Untuk melihat berapa baris yang kosong/NaN
df.isnull().sum()
```

[35]:
```
city                        0
city_development_index      0
gender                   3863
relevent_experience         0
enrolled_university         0
education_level             0
major_discipline         2222
experience                  0
company_size             5310
company_type             5476
last_new_job                0
training_hours              0
target                      0
dtype: int64
```

[31]:
```
# Menyiapkan template canvas
fig, ax = plt.subplots()

# Data asli
df['experience'].hist(bins=50, ax=ax, color='red', grid=False)

# Kita gunakan parameter alpha agar histogram df_cca menjadi sedikit transparan
df_cca['experience'].hist(bins=50, ax=ax, color='blue', alpha=0.6, grid=False)

# Menyiapkan legenda
label = ['Tanpa CCA', 'Dengan CCA']

# membuat judul plot dan menampilkannya
ax.set_title('Perbandingan histogram sebelum dan sesudah CCA')
ax.legend(label, loc=0)
plt.show()
```



Terakhir kita lakukan imputasi pada dara yang bernilai di Atas 0,5 % - 50 %, jika data numerik kita akan isikan nilai *median* sedangkan klasifikasi kita isikan *Missing*.

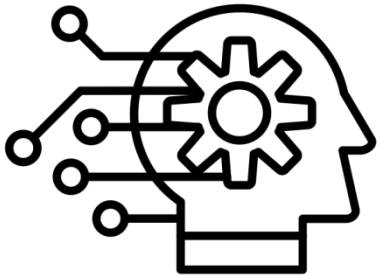
[43]:
```
# Membuat Pipeline untuk preprocessing
preprocessor_numerik = Pipeline([
    ('imputasi', SimpleImputer(strategy='median')),
    ('scaling', MinMaxScaler())
])

preprocessor_kategori = Pipeline([
    ('imputasi', SimpleImputer(strategy='constant', fill_value='missing')),
    ('encoding', OneHotEncoder(drop='first', sparse_output=False, handle_unknown='ignore'))
])

# Menggabungkan kedua pipeline di atas
preprocessor = ColumnTransformer([
    ('preprocessing numerik', preprocessor_numerik, var_numerik),
    ('preprocessing kategori', preprocessor_kategori, var_kategori)
])
```

Machine Learning Modeling

```
[36]: # Mendefinisikan variabel dependen (y) dan variabel independen (X)
      X = df.drop([target], axis=1)
      y = df[target]
```

Saatnya membuat prediksi dari data yang telah dibersihkan dan ditransformasi. Untuk prediksi data sudah dipisah berdasarkan fitur dan target, kolom target menjadi target (y) sedangkan kolom lainnya akan menjadi fitur atau (X). Algoritma yang digunakan ialah **Gradient Boosting dan XGBoosting** metode supervised learning.

Untuk bisa mengevaluasi performa dari model klasifikasi, pertama kita bisa mulai membuat *confusion matrix*.
Confusion matrix adalah sebuah matriks yang menunjukkan berapa banyak nilai data (baris data) yang diprediksi dengan tepat dan mana yang diprediksi dengan salah.



```
[233]: # Menghitung metrics klasifikasi satu per satu
       print('Nilai akurasi: {:.2f}'.format(accuracy_score(y_test, pred_test)))
       print('Nilai presisi: {:.2f}'.format(precision_score(y_test, pred_test)))
       print('Nilai recall: {:.2f}'.format(recall_score(y_test, pred_test)))
       print('Nilai f1: {:.2f}'.format(f1_score(y_test, pred_test)))
       print('Nilai AUC: {:.2f}'.format(roc_auc_score(y_test, mod_logreg.predict_proba(X_test)[:,1])))

       Nilai akurasi: 0.78
       Nilai presisi: 0.59
       Nilai recall: 0.37
       Nilai f1: 0.45
       Nilai AUC: 0.80
```

10.3 Akurasi

Akurasi adalah metrik yang paling sederhana dan intuitif. Akurasi menggambarkan seberapa sering model membuat prediksi yang benar.

Akurasi = (Jumlah Prediksi Benar) / (Jumlah Prediksi Benar + Jumlah Prediksi Salah)

10.4 Presisi

Presisi menggambarkan seberapa baik model mengidentifikasi kelas positif. Presisi didefinisikan sebagai:

Presisi = (True Positives) / (True Positives + False Positives)

10.5 Recall

Recall menggambarkan seberapa baik model mengidentifikasi semua kelas positif yang sebenarnya.

Recall = (True Positives) / (True Positives + False Negatives)

10.6 F1-Score

F1-Score adalah metrik yang menggabungkan presisi dan recall menjadi satu skor tunggal. F1-Score didefinisikan sebagai:

F1-Score = 2 * (Presisi * Recall) / (Presisi + Recall)

10.7 ROC AUC

ROC AUC mengukur kinerja model dalam mengklasifikasikan kelas positif dan negatif. ROC AUC memiliki rentang nilai antara 0 dan 1, dengan 1 mengindikasikan kinerja yang sempurna.

Adapun model pada akurasi dan AUC digunakan pada data training adalah Gradient Boosting dan XGBoots karena dikenal bagus untuk data classifikasi.

Sebelum Tuning

```
[585]: # Melihat hasil cross validation berdasarkan akurasi
       cv_akurasi
```

[585]:

	model	Train_Mean	std	Test_Score
0	gb	0.80	0.01	0.80
1	xgb	0.79	0.01	0.79

```
[586]: # Melihat hasil cross validation berdasarkan akurasi
       cv_auc
```

[586]:

	model	Train_Mean	std	Test_Score
0	gb	0.80	0.01	0.81
1	xgb	0.79	0.01	0.80

Setelah Tuning

```
[613]: # Melihat performa tuning berdasarkan akura
       grid_akurasi_urut
```

[613]:

	model	Training	Testing
0	xgb	0.836375	0.801832
1	gb	0.804524	0.796558

Perlu diperhatikan bahwa skor akurasi di kolom *Tr*

```
[615]: # Melihat performa tuning berdasarkan AUC
       grid_auc_urut
```

[615]:

	model	Training	Testing
0	xgb	0.869856	0.807348
1	gb	0.817549	0.804885

[621]:
```
# Kita gabungkan semua sebagai 'model_best'
model_best_estimator = pd.DataFrame({'model':daftar_nama_model, 'Param':best_estimator, 'Testing':auc_tuning_test})
model_best_estimator = model_best_estimator.sort_values(by='Testing', ascending=False, ignore_index=True)
model_best_estimator
```

[621]:

	model	Param	Testing
0	xgb	(ColumnTransformer(transformers=[('preprocessi...	0.807348
1	gb	(ColumnTransformer(transformers=[('preprocessi...	0.804885

Evaluasi

Deployment

Tools yang digunakan

NumPy

+ableau®

Terima kasih