Python 模組整理

* Scikit-learn (sklearn)
* datasets

‧load\_iris()

* linear\_model

‧LinearRegression()

* fit(X, y)
* predict()
* score(X, y) # accuracy
* coef\_
* intercept\_

‧LogisticRegression()

* fit(X, y)
* predict()
* score(X, y) # accuracy
* coef\_
* intercept\_
* feature\_selection

‧f\_regression(X, y) # f\_regression(X, y)[1] 是p-value

* preprocessing

‧LabelEncoder() # 創造 dummy variables

* fit\_transform()
* tree

‧DecisionTreeClassifier()

* fit(X, y)
* predict()
* plot\_tree(model.fit, feature\_names=, filled=True)
* feature\_importance\_
* model\_selection

‧train\_test\_split(X, y, test\_size=)

* metrics

‧accuracy\_score(y, y\_predicted)

‧classification\_report(y, y\_predicted)

‧plot\_confusion\_matrix(model, test\_X, test\_y, cmap=’pink\_r’)

‧silhouette\_score(X, labels) # D24

‧roc\_curve(y, y\_predicted)

‧auc(fpr, tpr)

* neighbors

‧KNeighborsClassifier()

* fit(X, y)
* predict()
* cluster

‧KMeans(n\_clusters=3)

* fit(X, y)
* labels\_

‧AgglomerativeClustering(linkage='ward', affinity='euclidean', n\_clusters=3) # hierarchical

* fit(X, y)
* labels\_
* ensemble

‧BaggingClassifier(n\_estimators = 100)

* fit(X,y)
* predict()

‧AdaBoostClassifier(n\_estimators = 100)

* fit(X,y)
* predict()

‧RandomForestClassifier(n\_estimators = 100)

* fit(X,y)
* predict()
* feature\_importance\_
* svm

‧SVC()

* fit(X,y)
* predict()
* Seaborn (sns)
* pairplot(df, hue=’’, palette=’inferno’)
* boxplot(x=, y=, data=, palette=);
* catplot(x='', y='', data=df, kind='box', col='sex', palette='inferno')
* heatmap(df.corr(), annot=True, cbar=True, cmap=’Oranges’, fmt=’.3f’, vmin=, vmax=)
* scatterplot(x=’’, y=’’, size=’’, alpha=1, data=, sizes=(1,500))
* set\_style(‘ticks’) # darkgrid, dark, whitegrid, white, ticks
* countplot(ax=, data=, x='', hue='', palette='');
* Numpy (np)
* array([])
* mean()
* nanmedian() # 忽略空值，計算中位數
* where(??)
* arrange(n1, n2) # array of [n1, n1+1, n1+2, …, n2]
* zeros()
* Pandas (pd)
* read\_csv(‘data route’)
* DataFrame(index=, data=, columns=[‘names’])

‧astype('int64')

‧apply(lambda x: )

‧agg({'review length':'sum' }) # ??

‧at[index, ‘column’] # 填入指定字串

‧corr()

‧columns.tolist()

‧copy()

‧describe()

‧drop([‘column name’], axis=, inplace=True)

‧dropna()

‧drop\_duplicates(subset='', inplace=True, keep='first')

‧fillna(df['sentiment'])

‧groupby(‘column’, as\_index=False)

‧head()

‧isnull()

‧isna()

‧index

‧info()

‧loc[]

‧map() # wordcloud analysis

‧plot(kind=, range=(n, n), bins=, edgecolor=, color=)

‧reindex()

‧reset\_index()

‧rolling()

‧size()

‧shift()

‧style.background\_gradient(cmap='autumn', subset=['Rating'])

‧sort\_values(‘column’, ascending=False)

‧split("-", n = 1, expand = True)

‧tail()

‧transpose()

‧unique()

‧values

‧value\_counts(sort=True)

* to\_datetime(df[‘column], errors=’coerce’)
* get\_dummies(‘column’, drop\_first=True) # converting categorical string data to numerics
* Matplotlib.pyplot (plt)
* Style

‧use(‘ggplot’) # 圖表背景樣式

‧available

* bar(x, y)
* scatter(x\_data, y\_data, color=’’)
* plot(x\_data, y\_data, color=’’, linewidth=, marker=, markersize=)
* figure(figuresize=(n,n), dpi=n)
* subplots(n, n, figsize=(n, n), sharey=False);

‧fig.suptitle(‘總標題’)

‧ax.set\_title(‘次標題’)

‧ax.twinx()

‧ax.plot(x, y, color=, label=, lw=)

‧ax.tick\_params(axis=’y’)

‧ax.set\_yscale(‘log’)

‧ax.get\_legend\_handles\_labels()

‧ax.legend(lines, labels, loc=0)

‧ax.pie(size\_cal, labels = label\_cal, startangle=90, shadow=0,

autopct='%1.2f%%', colors=['#228B22', '#CC0000', '#00BFFF'])

* title()
* xlabel()
* ylabel()
* imshow(pic, interpolation='bilinear') # wordcloud
* rcParams['figure.figsize']=(20, 10) # wordcloud
* tight\_layout(pad=0) # wordcloud
* axis(‘off’) # wordcloud
* show()
* Re
* sub(‘re語法’, ‘代換字串’, 字串變數)
* nltk # for stopwords
* corpus.stopwords.words(‘english’)
* wordcloud
* WordCloud(stopwords=, width=, height=, background\_color=,

mode=’RGB’, mask=, contour\_width=1, contour\_color='red', max\_words=, random\_state=1, colormap=, )

‧generate(text)

* STOPWORDS
* ImageColorGenerator
* PIL
* Image

‧open(‘pic route’) # wordcloud mask

* Plotly
* express (px)

‧choropleth(fig\_df, locations = 'Reviewer\_Location', locationmode='country

names', color = 'Branch', color\_continuous\_scale = 'Spectral', hover\_data=['Branch'], title = 'Country - Reviews')

‧update(layout\_coloraxis\_showscale=False)

‧show()

* graph\_objects (go)

‧Choropleth(locations = fig\_df['processed\_location'],

z = fig\_df['polarity\_cat'],

text = fig\_df['processed\_location'],

colorscale = 'Blues',

autocolorscale=False,

reversescale=True,

marker\_line\_color='darkgray',

marker\_line\_width=0.5)

‧Figure()

‧update\_layout(title\_text='Country - Reviews',

geo = dict(showframe=False,

showcoastlines=False,

projection\_type='equirectangular'),

annotations = [dict(

x=0.55,

y=0.1,

xref='paper',

yref='paper',

showarrow = False))

* Os
* walk(“folders’ direction”)
* path.join(dirname, filename)
* Scipy
* stats

‧spearmanr(x, y, nan\_policy=’omit’)

* Yfinance (yf)
* Ticker(‘stock\_name’)
* history(start=, end=)
* textblob
* TextBlob()

‧sentiment

* solarity
* subjectivity
* Torch
* Tensorflow