ANZ synthesized transaction
(Task 2)

By: Atefeh Gholipour

Project Description

	A	B	C	D	E	F	G	H	I	J	K
1	status	card_pres	bpay_	account	currency	long_lat	txn_descr	merchant_	merchant_	first_nam	balance
2	authorize	1		ACC-1598	AUD	153.41 -27	POS	81c48296-73be-44a7-		Diana	35.39
3	authorize	0		ACC-1598	AUD	153.41 -27	SALES-PO	830a451c-316e-4a6a-		Diana	21.2
4	authorize	1		ACC-1222	AUD	151.23 -33	POS	835c231d-8cdf-4e96-		Michael	5.71
5	authorize	1		ACC-1037	AUD	153.10 -27	SALES-PO	48514682-c78a-4a88-		Rhonda	2117.22
6	authorize	1		ACC-1598	AUD	153.41 -27	SALES-PO	b4e02c10-0852-4273-		Diana	17.95

	L	M	N	O	P	Q	R	S	T	U	V	W
1	date	gend	age	merchant_	merchant_	extraction	amount	transactio	country	customer_	merchant_	movemen
2	8/1/2018	F	26	Ashmore	QLD	2018-08-0	16.25	a623070bf	Australia	CUS-2487	153.38 -27	debit
3	8/1/2018	F	26	Sydney	NSW	2018-08-0	14.19	13270a2a	Australia	CUS-2487	151.21 -33	debit
4	8/1/2018	M	38	Sydney	NSW	2018-08-0	6.42	feb79e7e	Australia	CUS-2142	151.21 -33	debit
5	8/1/2018	F	40	Buderim	QLD	2018-08-0	40.9	2698170d	Australia	CUS-1614	153.05 -26	debit
6	8/1/2018	F	26	Mermaid	QLD	2018-08-0	3.25	329adf798	Australia	CUS-2487	153.44 -28	debit

Objective:

- Build a simple regression model to predict the annual salary for each customer

Dataset:

- Contains synthesised transaction of 3 months for 100 customers including inter bank, payment, phone bank, POS, sales-POS and salary transactions.

[7]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12043 entries, 0 to 12042
Data columns (total 23 columns):
status               12043 non-null object
card_present_flag    7717 non-null float64
bpay_biller_code     885 non-null object
account              12043 non-null object
currency             12043 non-null object
long_lat             12043 non-null object
txn_description      12043 non-null object
merchant_id          7717 non-null object
merchant_code        883 non-null float64
first_name           12043 non-null object
balance              12043 non-null float64
date                 12043 non-null datetime64[ns]
gender               12043 non-null object
age                  12043 non-null int64
merchant_suburb      7717 non-null object
merchant_state       7717 non-null object
extraction           12043 non-null object
amount               12043 non-null float64
transaction_id       12043 non-null object
country              12043 non-null object
customer_id          12043 non-null object
merchant_long_lat    7717 non-null object
movement             12043 non-null object
dtypes: datetime64[ns](1), float64(4), int64(1), object(17)
memory usage: 2.1+ MB
```

Analysis Steps:
Step 1: Preliminary analysis and feature selection

- Drop some irrelevant features
- Check for null data (there is no null in remaining dataset)

[10]:
```
# drop irrelevant columns

df = dataset.drop(['status', 'card_present_flag', 'bpay_biller_code',␣
↪'account', 'currency', 'merchant_id',
                   'merchant_code', 'first_name', 'date', 'merchant_suburb',␣
↪'merchant_state', 'extraction',
                   'transaction_id', 'country', 'merchant_long_lat',␣
↪'movement'], axis = 1)
```

[11]:
```
df.head()
```

[11]:

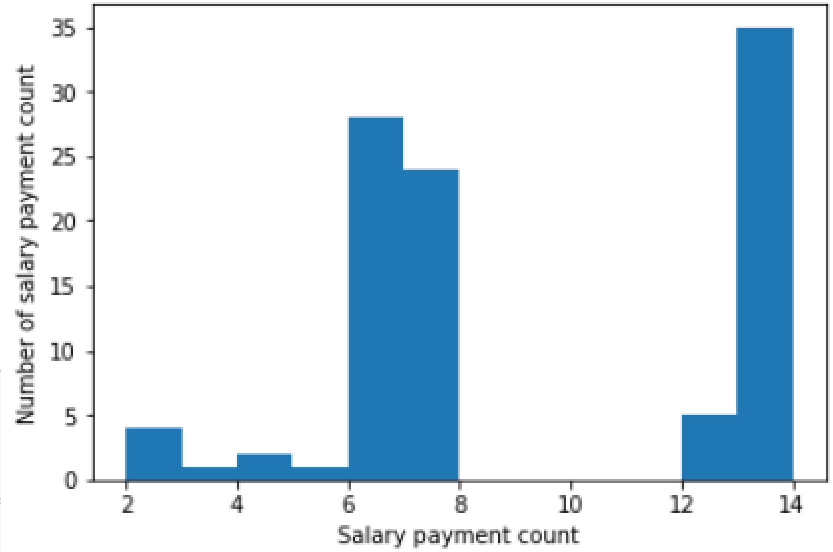
	long_lat	txn_description	balance	gender	age	amount	customer_id
0	153.41 -27.95	POS	35.39	F	26	16.25	CUS-2487424745
1	153.41 -27.95	SALES-POS	21.20	F	26	14.19	CUS-2487424745
2	151.23 -33.94	POS	5.71	M	38	6.42	CUS-2142601169
3	153.10 -27.66	SALES-POS	2117.22	F	40	40.90	CUS-1614226872
4	153.41 -27.95	SALES-POS	17.95	F	26	3.25	CUS-2487424745

[12]:
```
# check for nulls
df.isnull().sum()
```

[12]:
```
long_lat           0
txn_description    0
balance            0
gender             0
age                0
amount             0
customer_id        0
dtype: int64
```

Analysis Steps:
Step 2: Data Extraction (annual salary)

- Annual salary should be extracted from txn_description column
- Distribution of the salary of the customers are shown as:



- There is no data for 16-08-2018.
- For finding annual salary from 3 month payment three:

[20]:
```
# Calculate annual salary
# if Salary_Count >= 12 then payment is weekly
# if Salary_Count <= 5 then payment is monthly
# if other value then payment is fortnightly
```

[21]:
```
df_CI_S ['Annual_Salary'] = 0
for i in range (0, len(df_CI_S)):
    if int(df_CI_S.Salary_Count[i]) >= 12:
        df_CI_S.Annual_Salary[i] = df_CI_S ['PAY/SALARY'][i]/(df_CI_S.
 ↪Salary_Count[i]) / 7 * 356
    elif int(df_CI_S.Salary_Count[i]) <= 5:
        df_CI_S.Annual_Salary[i] = df_CI_S ['PAY/SALARY'][i]/(df_CI_S.
 ↪Salary_Count[i]) * 12
    else:
        df_CI_S.Annual_Salary[i] = df_CI_S ['PAY/SALARY'][i]/(df_CI_S.
 ↪Salary_Count[i]) / 14 * 356

# all transaction multipy 4 to obtain for one year

df_CI_S [['INTER BANK', 'PAYMENT', 'PHONE BANK', 'POS',
          'SALES-POS']] =4 * df_CI_S [['INTER BANK','PAYMENT', 'PHONE BANK',␣
 ↪'POS', 'SALES-POS']]
```

Analysis Steps:
Step 2: Data Extraction (customers' location)

- State of each customer are found based on latitude and longitude information.
- There is one out-of-range latitude. So, remove it (row number 2036).

```
# find state of each customer base on longitude and latitude information

import geopy
from geopy.geocoders import Nominatim
def LoctoState(LOC):
    locator = Nominatim(user_agent="myGeocoder")
    coordinates = LOC
    location = locator.reverse(coordinates, exactly_one = True, timeout = 10)
    address = location.raw['address']
    state = address.get('state','')
    return state
#print (LoctoState( "-27.51,153.03"))

df_S_B['state'] = ''
for i in range (0, len(df_S_B)):
    df_S_B['state'][i] = LoctoState(df_S_B['lat_long'][i])
```

Analysis Steps:
Step 2: Data Extraction

- Attribute 'total spend' is defined which includes inter bank, payment, phone bank, POS and sales-POS.
- Initial balance of each account has been added.

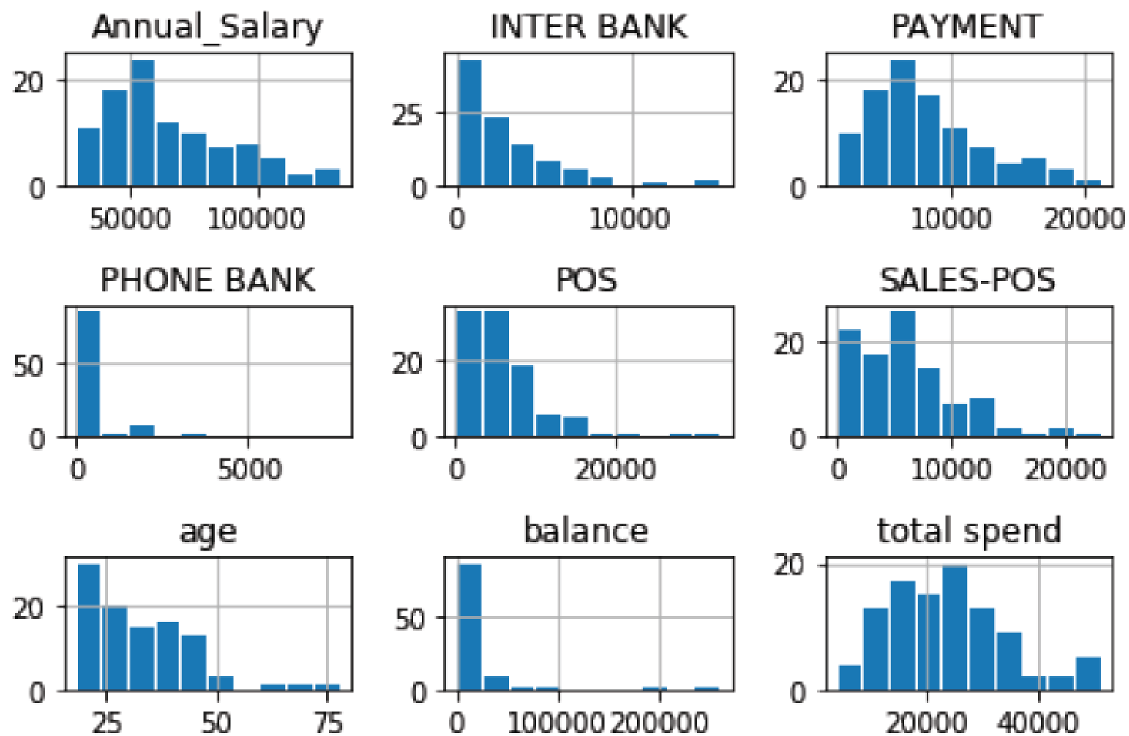
- The final dataset for analysis is as below:

	age	gender	INTER BANK	PAYMENT	PHONE BANK	POS	SALES-POS	Annual_Salary	balance	total spend	state
0	53	F	0	5184	2184	2992.04	4251.40	49355	463.96	14611.44	Queensland
1	21	M	4004	15828	0	2425.48	13477.80	90999	2335.35	35735.28	Western Australia
2	28	M	1080	3408	0	5425.88	12132.28	48734	823.53	22046.16	Victoria
3	34	F	1000	10388	0	8249.24	7293.76	87036	1726.28	26931.00	New South Wales
4	34	F	3068	12068	0	9222.60	10539.84	99266	12529.59	34898.44	South Australia

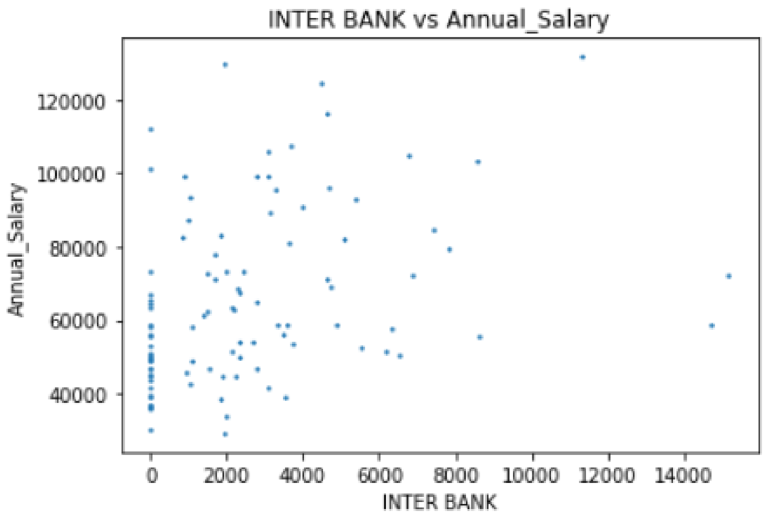
Analysis Steps:
Step 3: Data visualisation

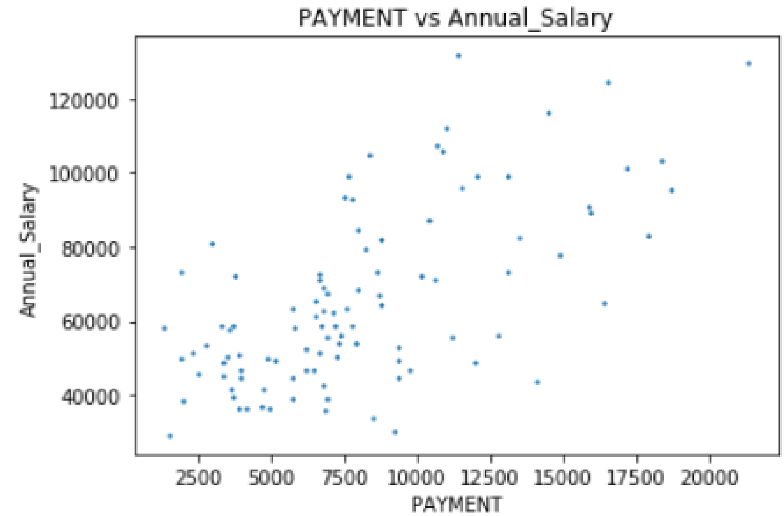
- Distribution of features

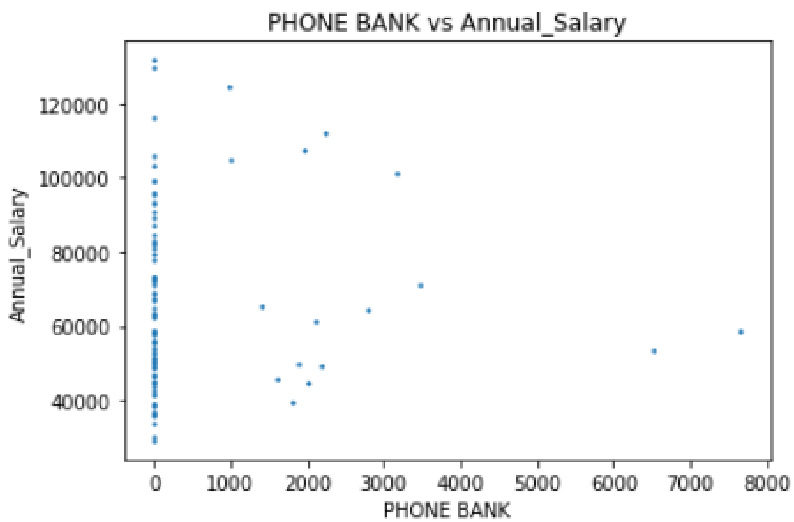
[25]:
```
# Histogram of Data visulisation
df_F.hist(rwidth = 0.9)
plt.tight_layout()
```

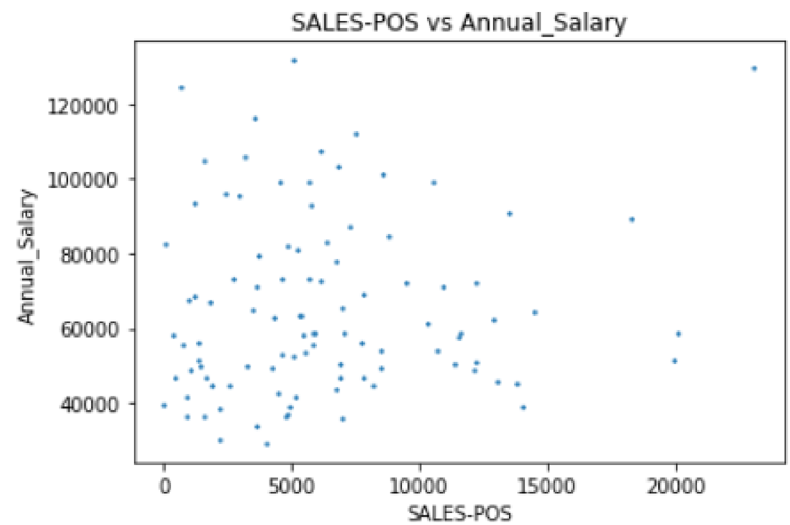


Analysis Steps:
Step 3: Data visualisation (continuous features)

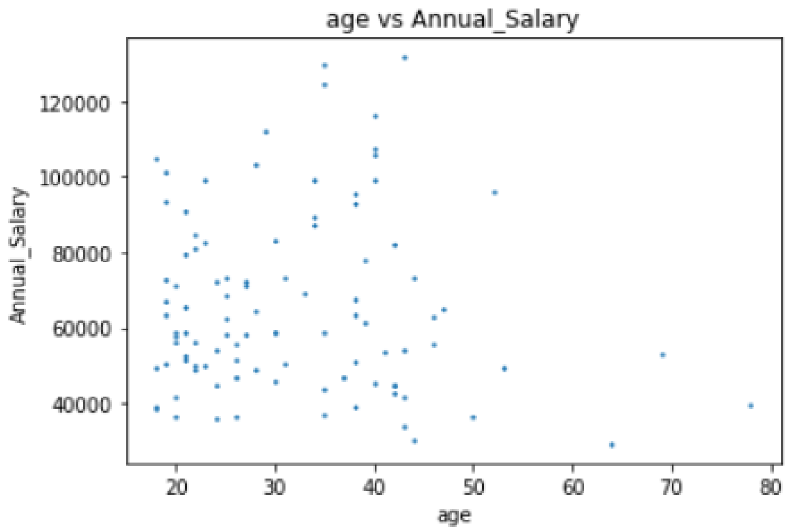


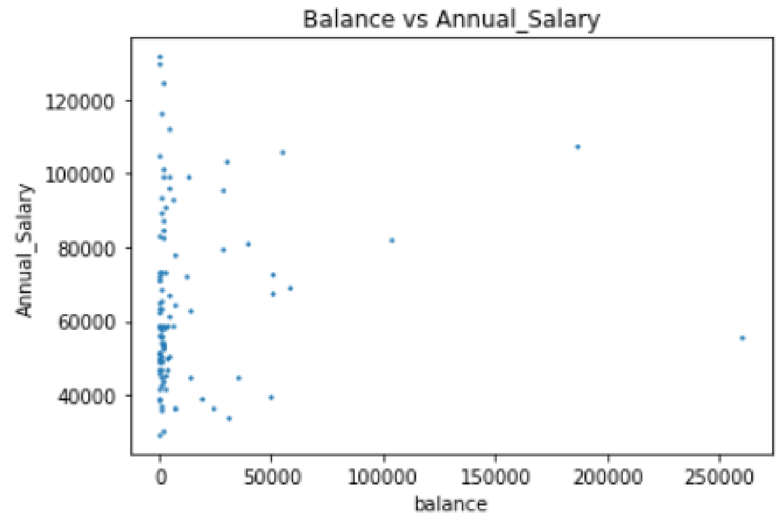


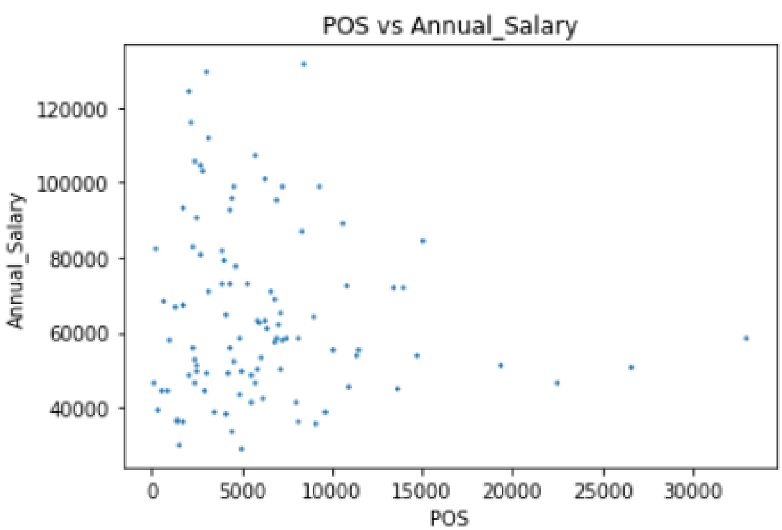


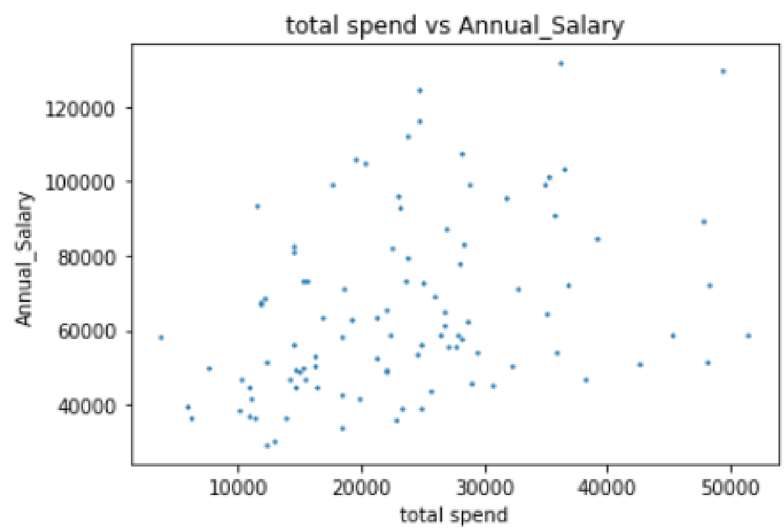


Analysis Steps:
Step 3: Data visualisation (continuous features)

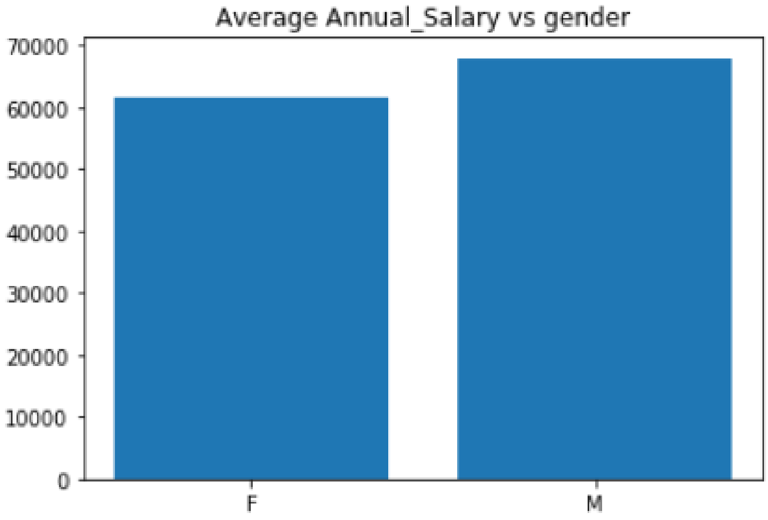


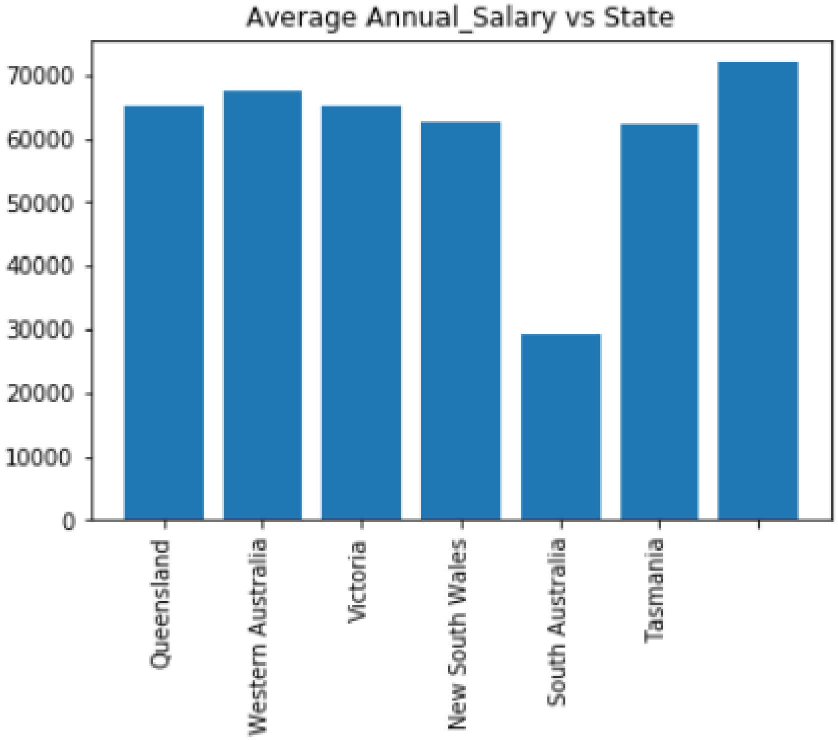






Analysis Steps:
Step 3: Data visualisation (categorical features)





Analysis Steps:
Step 4: Feature selection (correlation calculation)

- Drop features with lower correlation

```
# Check linearity using correlation coeficient matrix
correlation = df_F[['Annual_Salary', 'age', 'INTER BANK', 'PAYMENT', 'PHONE␣
↪BANK', 'POS', 'SALES-POS',
                   'balance', 'total spend']].corr()
print(correlation)
```

```
              Annual_Salary       age  INTER BANK   PAYMENT  PHONE BANK  \
Annual_Salary      1.000000 -0.061377    0.352362  0.639631    0.033414
age               -0.061377  1.000000   -0.099233  0.026884    0.103961
INTER BANK         0.352362 -0.099233    1.000000  0.087386   -0.081680
PAYMENT            0.639631  0.026884    0.087386  1.000000   -0.132095
PHONE BANK         0.033414  0.103961   -0.081680 -0.132095    1.000000
POS               -0.086938 -0.036929    0.181437 -0.123618   -0.052313
SALES-POS          0.100400 -0.139284    0.158792  0.121610    0.007496
balance            0.110321  0.237992    0.211241  0.018268    0.026537
total spend        0.371378 -0.086176    0.476295  0.416710    0.015633

                    POS  SALES-POS   balance  total spend
Annual_Salary -0.086938   0.100400  0.110321     0.371378
age           -0.036929  -0.139284  0.237992    -0.086176
INTER BANK     0.181437   0.158792  0.211241     0.476295
PAYMENT       -0.123618   0.121610  0.018268     0.416710
PHONE BANK    -0.052313   0.007496  0.026537     0.015633
POS            1.000000   0.418105 -0.000239     0.689232
SALES-POS      0.418105   1.000000 -0.153300     0.756790
balance       -0.000239  -0.153300  1.000000     0.002437
total spend    0.689232   0.756790  0.002437     1.000000
```

Analysis Steps:
Step 4: Feature selection (f_regression)

- get_dummies for categorical variables and StandardScaler for continuous variables
- Use f_regression to calculate F-score and P-value
- Feature with lower F_score has been removed

```python
# feature selection
from sklearn.feature_selection import f_regression
result = f_regression(X,Y)
f_score = result [0]
p_value = result [1]

columns = list(X.columns)
print("  Feature   ", "                    F_score   " , "                  P_value   ")
print("  -------                      -------                    ------- ")
for i in range(0, len(columns)):
    f1=  f_score[i]
    p1=  p_value[i]
    print(" ", columns[i].ljust(20),"   " , f1, "   ", p1)
```

```
 Feature                         F_score                      P_value
 -------                        -------                      -------
 INTER BANK               13.077682186681905    0.00047643710935370927
 PAYMENT                  68.01558513144904    7.985674327460962e-13
 SALES-POS                0.9104035219992912    0.34237914303848604
 balance                  1.2633164571082347    0.26379816386664495
 total spend              14.845159693260255    0.0002095941070946692
 gender_M                 1.7205564326030651    0.19271752801591077
 state_New South Wales     0.3735928480105821    0.5424821000548368
 state_Queensland         0.0010899400623479772    0.9737310705621645
 state_South Australia     0.07316966142052807    0.7873511721337136
 state_Tasmania           2.350633544897465    0.12848730409545903
 state_Victoria           0.8304095476594371    0.3644132122518978
 state_Western Australia     1.337959517193049    0.25023486278067525
```

Analysis Steps:
Step 4: Linear Regression

- Final dataset is as follows:

	gender	INTER BANK	PAYMENT	Annual_Salary	total spend
0	F	0	5184	49355	14611.44
1	M	4004	15828	90999	35735.28
2	M	1080	3408	48734	22046.16
3	F	1000	10388	87036	26931.00
4	F	3068	12068	99266	34898.44

- After splitting data to test and train datasets and fitting the model:

```
score_test = LR.score(X_test, Y_test)
RMSE = math.sqrt(mean_squared_error(Y_test, Y_Predicted))

print('r2 score for Linear Regression is:' ,score_test)
print('RMSE for Linear Regression is:' ,RMSE)
```

```
r2 score for Linear Regression is: 0.6173311697006681
RMSE for Linear Regression is: 0.6155336134400503
```

- However some method such as KFold can slightly improve the r2_value.

```
# KFold
from sklearn.model_selection import KFold
folds = KFold(n_splits = 5, shuffle = True, random_state = 100)
scores = cross_val_score(LR, X_train, Y_train, scoring='r2', cv=folds)
scores
```

```
array([ 0.40019539,  0.68358956,  0.58267536, -0.19175429, -0.09101695])
```

Summary

- Even with using feature selection method, the selected feature can not leads to a good model and r2_value is not high enough (r2_score = 0.61).

- As the number of customer was very low, obtaining the model with low accuracy was predictable.