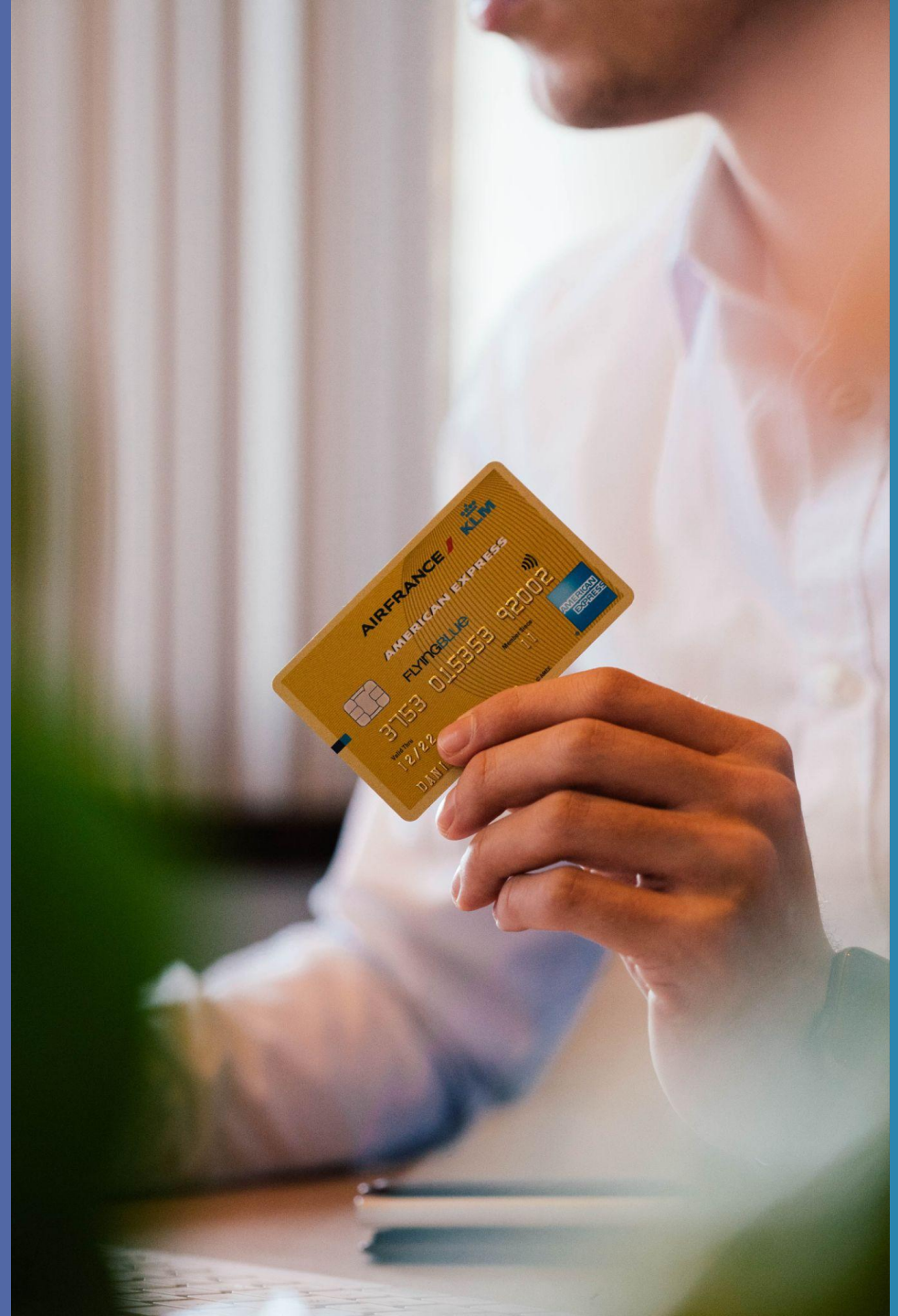


Credit Card Approval Prediction

Project 4
Team 7

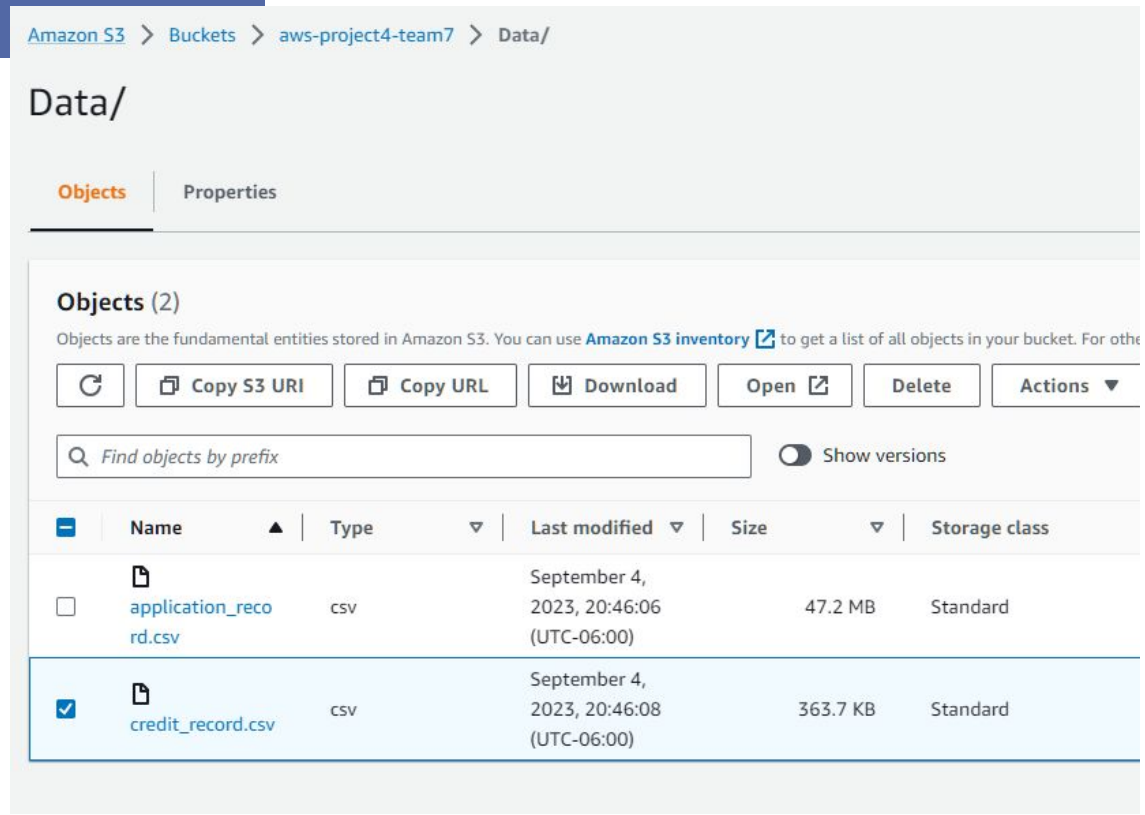
Our team

Erik

Pedro

Karen

Brian



AWS

Spark

```
# 1. Read in the AWS S3 bucket into a DataFrame.
from pyspark import SparkFiles
url1 = "https://aws-project4-team7.s3.amazonaws.com/Data/application_record.csv"
url2 = "https://aws-project4-team7.s3.amazonaws.com/Data/credit_record.csv"
spark.sparkContext.addFile(url1)
df1 = spark.read.csv(SparkFiles.get("application_record.csv"), sep=",", header=True, ignoreLea
spark.sparkContext.addFile(url2)
df2 = spark.read.csv(SparkFiles.get("credit_record.csv"), sep=",", header=True, ignoreLeadingW
```

Our data set

	ID	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	NAME_INCOME_TYPE	NAME_EDUCATION_TYPE	NAME_FAMILY_STATUS
0	5008804	M	Y	Y	0.0	427500.0	Working	Higher education	Civil marriage
1	5008805	M	Y	Y	0.0	427500.0	Working	Higher education	Civil marriage
2	5008806	M	Y	Y	0.0	112500.0	Working	Secondary / secondary special	Married

```
# Reading initial datasets
df = pd.read_csv('/content/sample_data/application_record.csv')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 438557 entries, 0 to 438556
Data columns (total 18 columns):
```

```
# Merging the 2 datasets with column ID. Reduced to 24320 samples.
df_t = df.merge(df_cr, how='inner', on='ID')
df_t.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 24320 entries, 0 to 24319
Data columns (total 14 columns):
```

ETL

Conversion Y/N to binary

Age & Years Employed transformation

```
#Changing in 'FLAG_OWN_CAR' to 1 and 0 for easier use.
for index, row in df_t.iterrows():
    if row['FLAG_OWN_CAR'] == 'N':
        df_t.at[index, 'FLAG_OWN_CAR'] = 0
    elif row['FLAG_OWN_CAR'] == 'Y':
        df_t.at[index, 'FLAG_OWN_CAR'] = 1
#Changing in 'FLAG_OWN_REALTY' to 1 and 0 for easier use.
for index, row in df_t.iterrows():
    if row['FLAG_OWN_REALTY'] == 'N':
        df_t.at[index, 'FLAG_OWN_REALTY'] = 0
    elif row['FLAG_OWN_REALTY'] == 'Y':
        df_t.at[index, 'FLAG_OWN_REALTY'] = 1
```

```
# Transforming column DAYS BIRTH into Age in years
for index, row in df_t.iterrows():
  df_t.at[index, 'DAYS_BIRTH'] = ((df_t.at[index, 'DAYS_BIRTH']*-1)/365.25).round()
```

Data Preprocessing

Transformation of Categorical columns to Numerical (creation of dummies)

Standardize values

```
# Choosing categorical columns that need to be transformed into dummy variables
columns_to_convert = [
    'GENDER', 'INCOME_TYPE', 'EDUCATION_TYPE',
    'FAMILY_STATUS', 'HOUSING_TYPE']
dummies = pd.get_dummies(df_t, columns=columns_to_convert)
```

```
# Create a StandardScaler instances
scaler = StandardScaler()
# Fit the StandardScaler
scaler.fit(X_train)
# Scale the data
X_train_scaled = scaler.transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

Model / Neural Network

```
# Define the model - deep neural net, i.e., the number of input features
nn = tf.keras.models.Sequential()
# First hidden layer
nn.add(tf.keras.layers.Dense(units=80, input_dim=30, activation='relu'))
# Second hidden layer
nn.add(tf.keras.layers.Dense(units=80, activation='relu'))
# Output layer
nn.add(tf.keras.layers.Dense(units=1, activation='sigmoid'))
# Check the structure of the model
nn.summary()
```

	feature	importance
4	AGE	0.261402
3	AMT_INCOME_TOTAL	0.241768
5	YEARS_EMPLOYED	0.154060
1	FLAG_OWN_PROPERTY	0.040484
6	CNT_FAM_MEMBERS	0.038621
0	FLAG_OWN_CAR	0.034008
2	CNT_CHILDREN	0.027355

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_6 (Dense)              (None, 100)               800

dense_7 (Dense)              (None, 100)               10100

dense_8 (Dense)              (None, 100)               10100

dense_9 (Dense)              (None, 1)                 101

_________________________________________________________________
```

Results

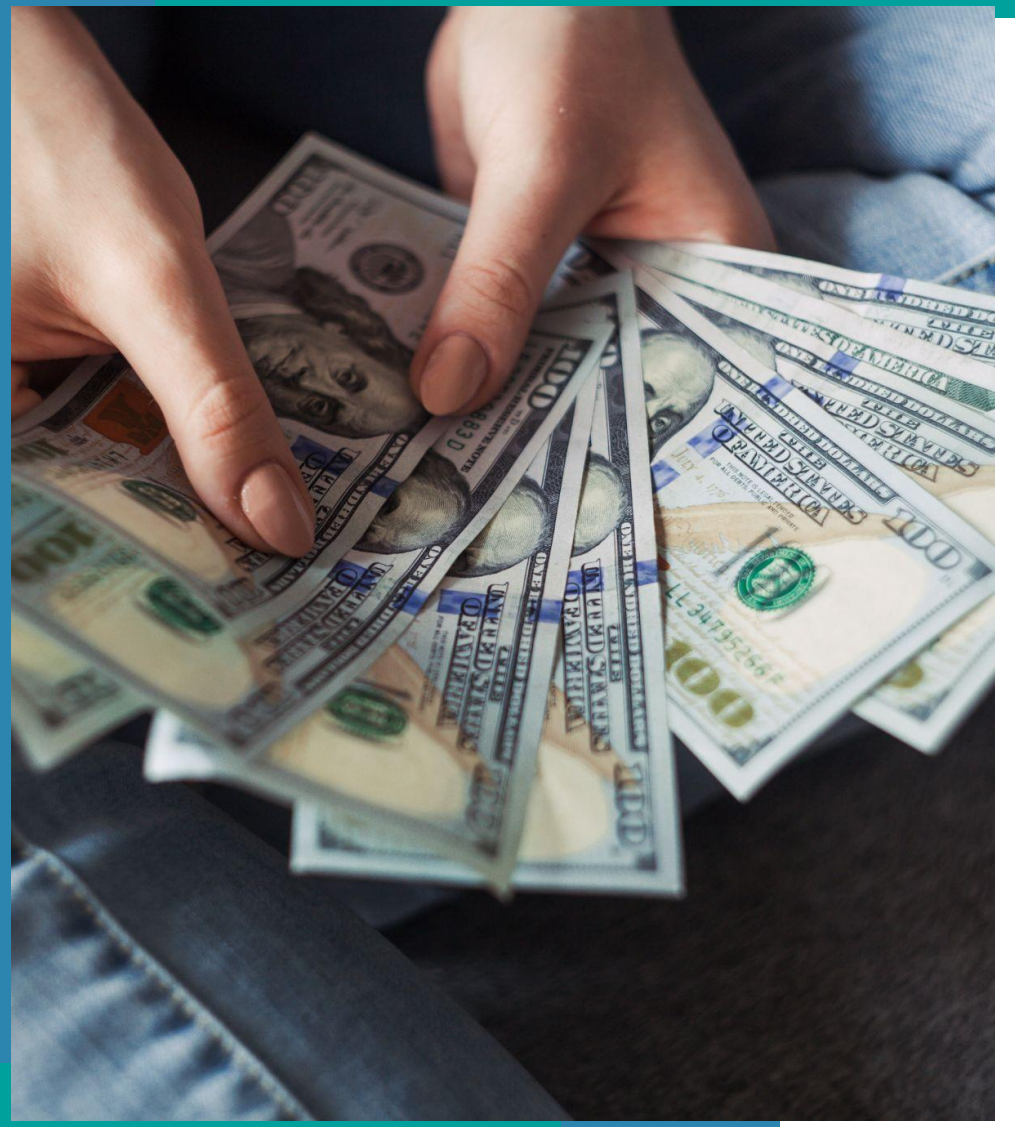
	Features	Hidden Layers	Epochs	Neurons	Optimizer	Activation F	Accuracy
Model 1	30	2	10	80	Adam	Relu	98.49%
Model 2	7	2	10	80	Adam	Relu	98.51%
Model 3	7	3	10	100	adamax	Tanh	98.51%
Model 4	7	2	10	100	adamax	Tanh / Relu	98.51%

Random Forest for Models 2,3 and 4

Highest Achieved Accuracy:
98.51%

Tableau Dashboards

Exploration of demographic trends



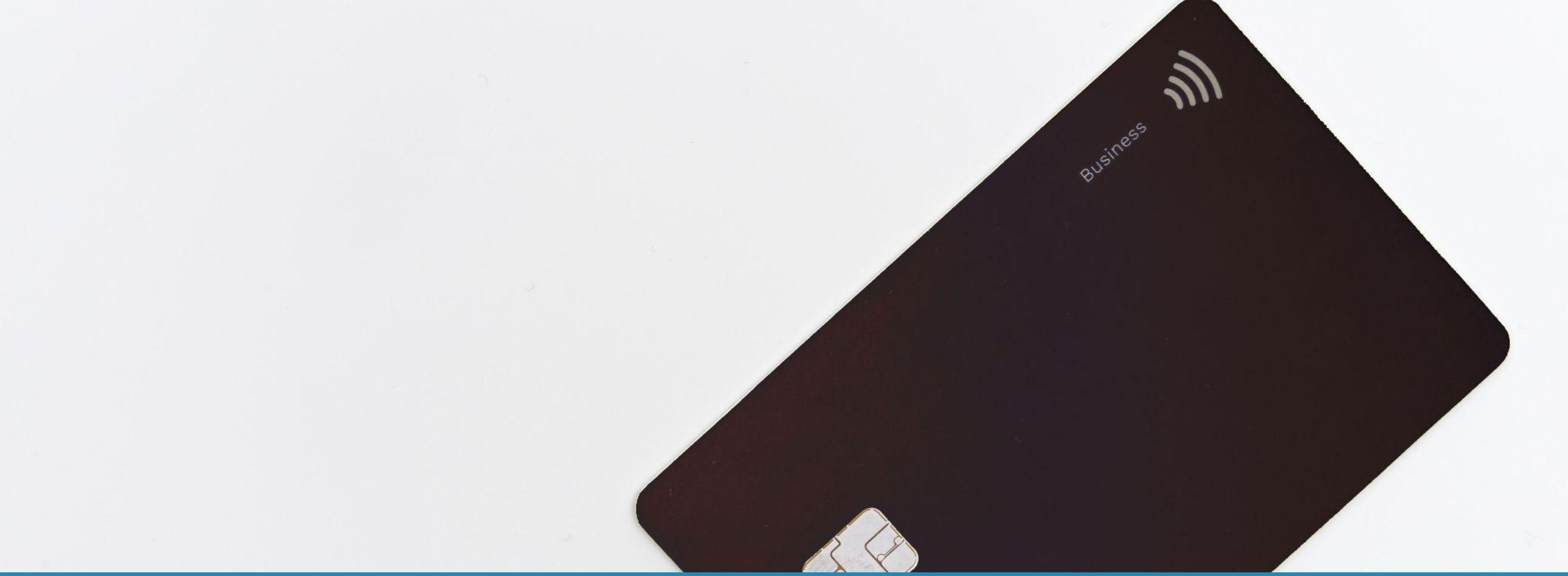
How is the income distributed by type and gender?

Are age & years of working experience a determining factor?

Does family size matter?

Conclusion

- We achieved good results with the highest accuracy being **98.51%**
- Our predictive power in assessing credit card approval **improved through the iteration**
- We find this model useful for bank/ companies that wish to assess the credit card approvement analysis



Thanks for your attention!

Questions?