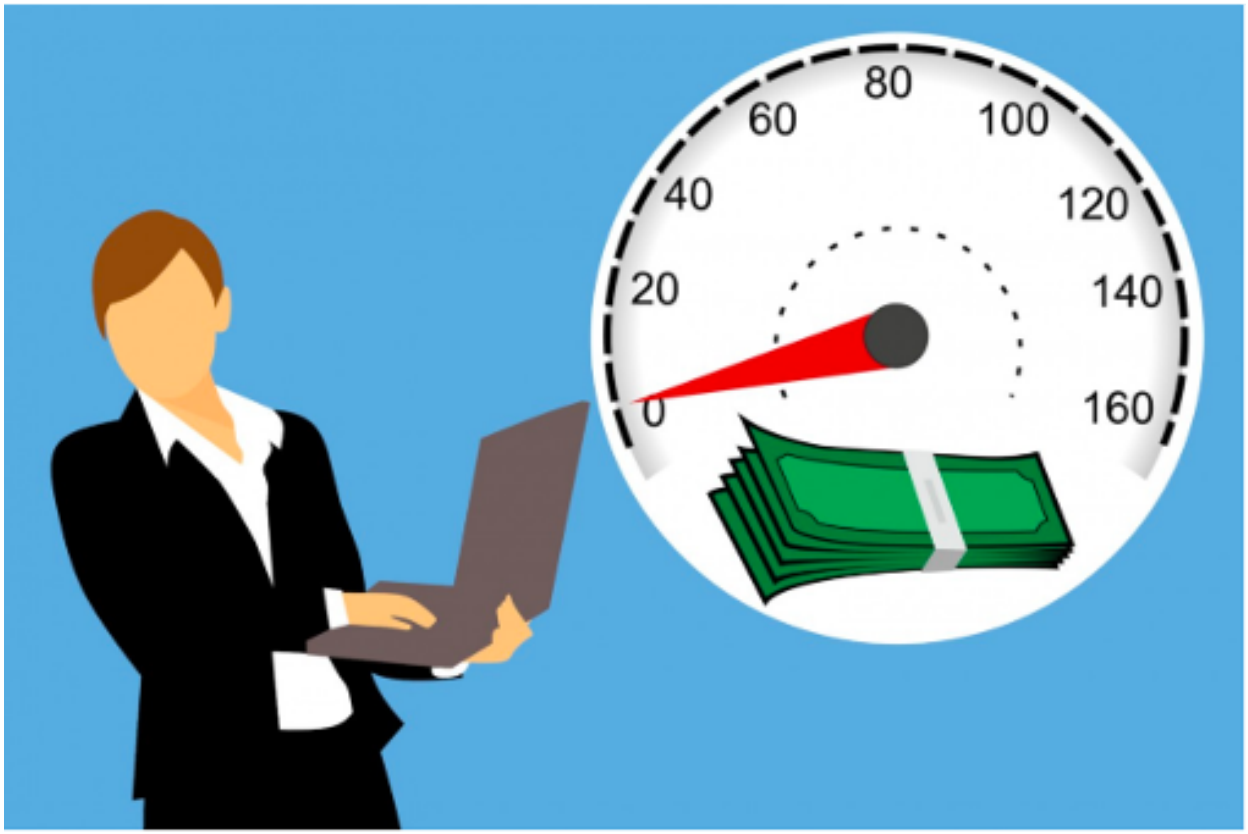
APPLIED MACHINE LEARNING

SALARY PREDICTION

A project report by Nihar Tripathi



❖ Use Case and Data Description

- **Project Motivation:** These days, people assess multiple factors before looking and applying for a job/internship. One of the major factors a person looks at is the salary given for their job position. Similarly, being a student I was recently looking for internship opportunities in the tech sector and one of the major factors I had in my mind was how much salary I could potentially get in different job roles looking at my qualifications. That inspired me and gave me an idea to go forth with this Salary Prediction project.

- **Use Case:** Multiple factors determine an individual's probability of getting a job with a good salary, some of which are the Educational qualifications, the workclass, the occupation they are applying for and hours they get to work. This salary prediction model can be used on professional networking sites to help individuals filter job salaries by filtering jobs according to their necessity.

- **Problem Description:** Predicting the salary of an individual to help determine whether an individual would be able to land a job that will pay them a salary of 50,000$ or not.

- **Data Collection and Extraction:** The dataset's name is 'Census Income Dataset' and is referenced from the UCI Machine Learning repository and the data is further extracted by Barry Becker for analysis.

    - The dataset consists of 32,561 data rows and 15 features for each data.

❖ About the Dataset

- ❖ The shape of the dataset is 32,561*15.
- ❖ The 15 columns depict the different features of the dataset.

salaryprediction

AGE	WORKCLASS	FNLWGT	EDUCATION	EDUCATIONNUM	MARITALSTATUS	OCCUPATION	RELATIONSHIP	RACE	SEX	CAPITALGAIN	CAPITALLOSS	HOURSPERWEEK	NATIVECOUNTRY	ABOVE50K
39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	0
50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	0
38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	0
53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States	0
28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba	0

- ❖ Dataset information:

```
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   AGE             32561 non-null  int64
 1   WORKCLASS       32561 non-null  object
 2   FNLWGT          32561 non-null  int64
 3   EDUCATION       32561 non-null  object
 4   EDUCATIONNUM    32561 non-null  int64
 5   MARITALSTATUS   32561 non-null  object
 6   OCCUPATION      32561 non-null  object
 7   RELATIONSHIP    32561 non-null  object
 8   RACE            32561 non-null  object
 9   SEX             32561 non-null  object
 10  CAPITALGAIN     32561 non-null  int64
 11  CAPITALLOSS     32561 non-null  int64
 12  HOURSPERWEEK    32561 non-null  int64
 13  NATIVECOUNTRY   32561 non-null  object
 14  ABOVE50K        32561 non-null  int64
```

- ❖ As seen above, out of the 15 features(columns), 7 features are of 'integer' data type and there are 8 columns of 'object' data type.
- ❖ As of now there are no 'null' values present in the dataset.

❖ Data Cleaning and Encoding

- ❖ Plotting the value counts for each feature, it was found that some of the features contained '?' values which did not infer anything related to the data.
- ❖ Features like WORKCLASS, OCCUPATION and NATIVECOUNTRY had such undefined values.

```
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
```

- ❖ These values were changed to 'NaN' which means 'Not a Number' which basically converted them to null values and then the rows having such values were dropped from the dataset.
- ❖ To encode the data, I created duplicate columns for all features having data type as 'object' as I did not want to meddle with the original dataset.

workclass_encoder	education_encoder	maritalStatus_encoder	occupation_encoder	relationship_encoder	race_encoder	sex_encoder	nativeCountry_encoder
6	13	3	12	0	2	1	15
1	13	6	5	1	2	1	15
0	15	2	11	0	2	1	15
0	7	6	11	1	1	1	15
0	13	6	7	4	1	0	8

- Further, I encoded these duplicate columns to change their values and data type to integer.

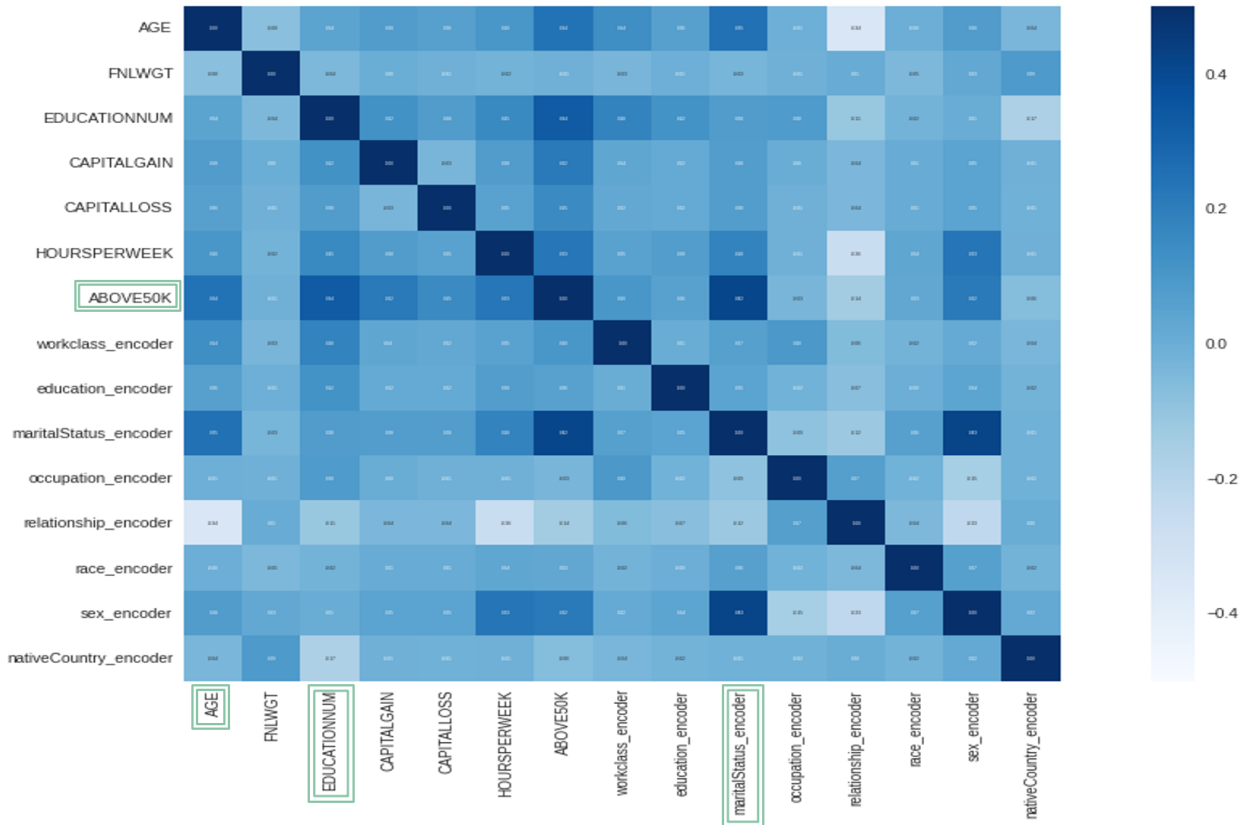
```
{' Wife': 0, ' Husband': 1, ' Other-relative': 2, ' Own-child': 3, ' Unmarried': 4, ' Not-in-family': 5}
```

	AGE	WORKCLASS	FNLWGT	EDUCATION	EDUCATIONNUM	MARITALSTATUS	OCCUPATION	RELATIONSHIP	RACE	SEX	...	NATIVECOUNTRY
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	...	United-States
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	...	United-States
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	...	United-States

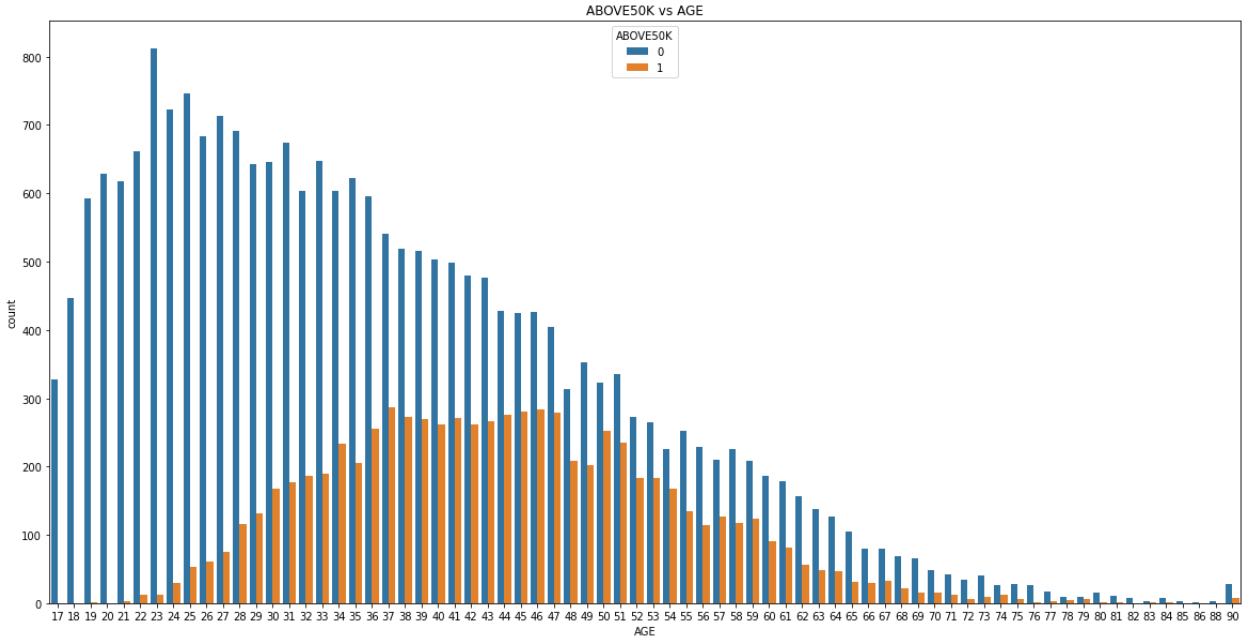
3 rows × 23 columns

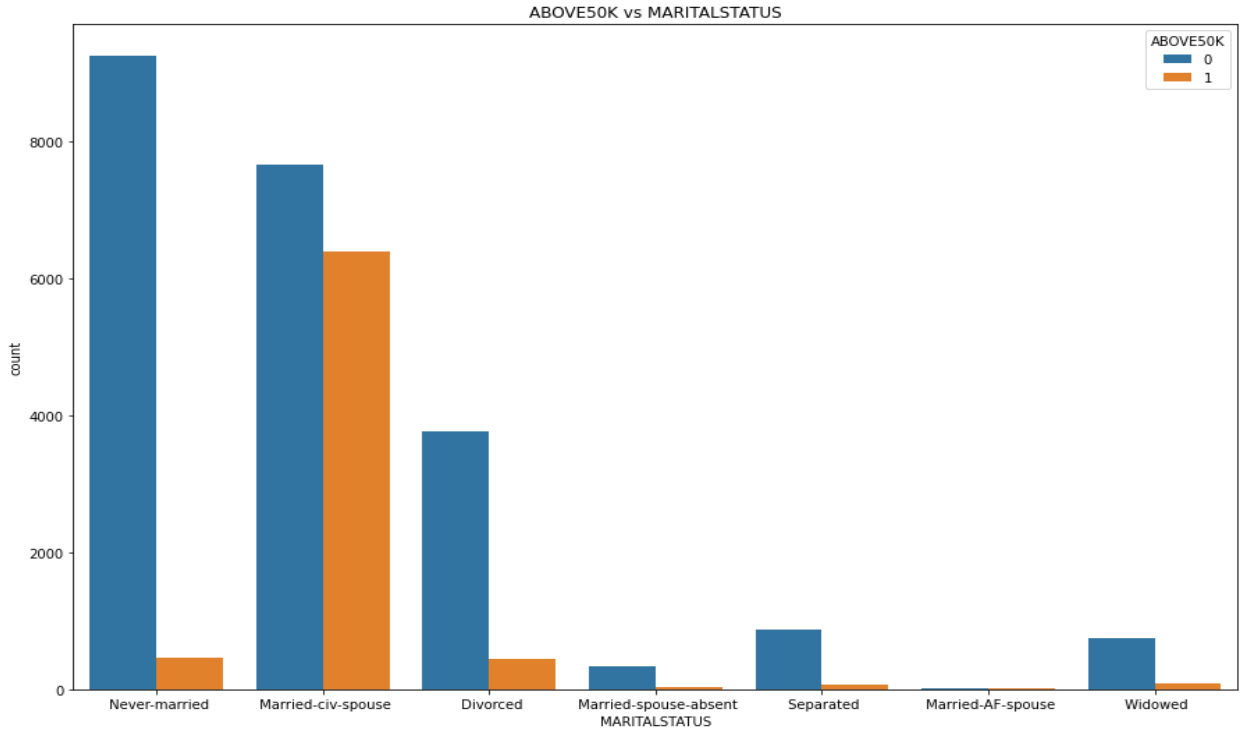
❖ Exploratory Data Analysis (EDA)

- ❖ Firstly, we decide our target variable which is 'ABOVE50K'.
- ❖ Then we find the correlation between features by plotting correlation values and correlation matrix.



- ❖ As you can see, features like AGE, EDUCATIONNUM and maritalStatus_encoder correlate the most with our target variable.
- ❖ **Plotting Relationship Graphs:** Plotting the relationship graphs of different features with respect to our target gave us a better insight of their relationships with each other.





PS: Other graphs are plotted in the code but not mentioned in the report. Pls refer to the code to look at them.

❖ Feature Scaling

- ❖ A temporary dataset is created using the initial ‘integer’ data type columns and the encoded columns.
- ❖ Feature Scaling is done to this temporary dataset using the StandardScaler method to remove the mean and scale each feature to unit variance.

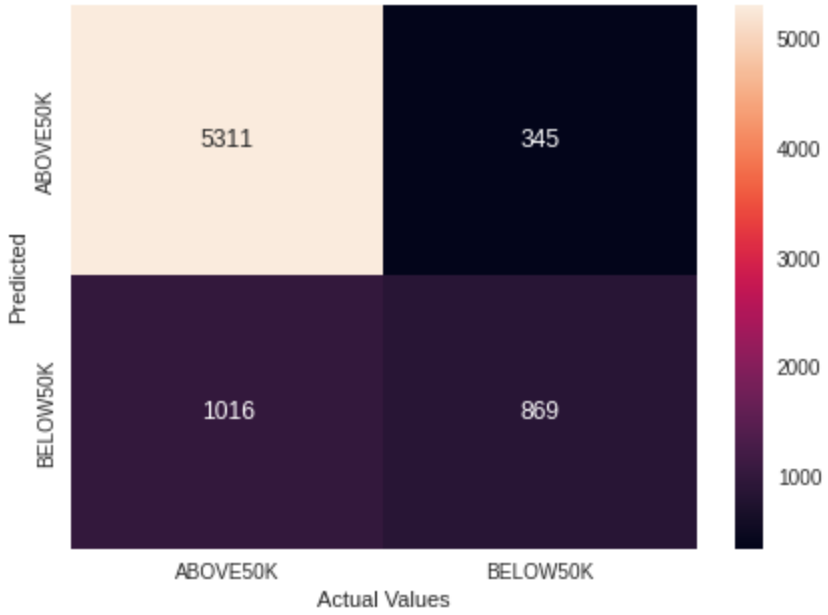
❖ Model-1: Logistic Regression

- ❖ On plotting the value counts of the target variable we find that there are 7508 training examples belonging to class 1 and 22654 examples belonging to class 0.
- ❖ Thereby we can say that our dataset is biased towards class 0.
- ❖ The logistic regression will be applied in two different ways - Performing Logistic Regression with taking equal number of samples of both classes & Performing Logistic Regression on the whole dataset.

➔ Logistic regression for the complete dataset:

- ◆ Here we are training the whole dataset for inference.
- ◆ Next, we use the sklearn library to split the dataset into two parts - training data and test data.
- ◆ For validation, there is no validation set created as I have used the cross validation method which randomly makes bins of around 20% of training data and then uses these bins to validate the training set.
- ◆ Then logistic regression is performed on the data with the following parameters : `logisticregression = LogisticRegression(C=100, max_iter=50, penalty-'l2', solver='liblinear').fit(X_train, y_train)  #c = 1/ lam`

- **Confusion matrix:**

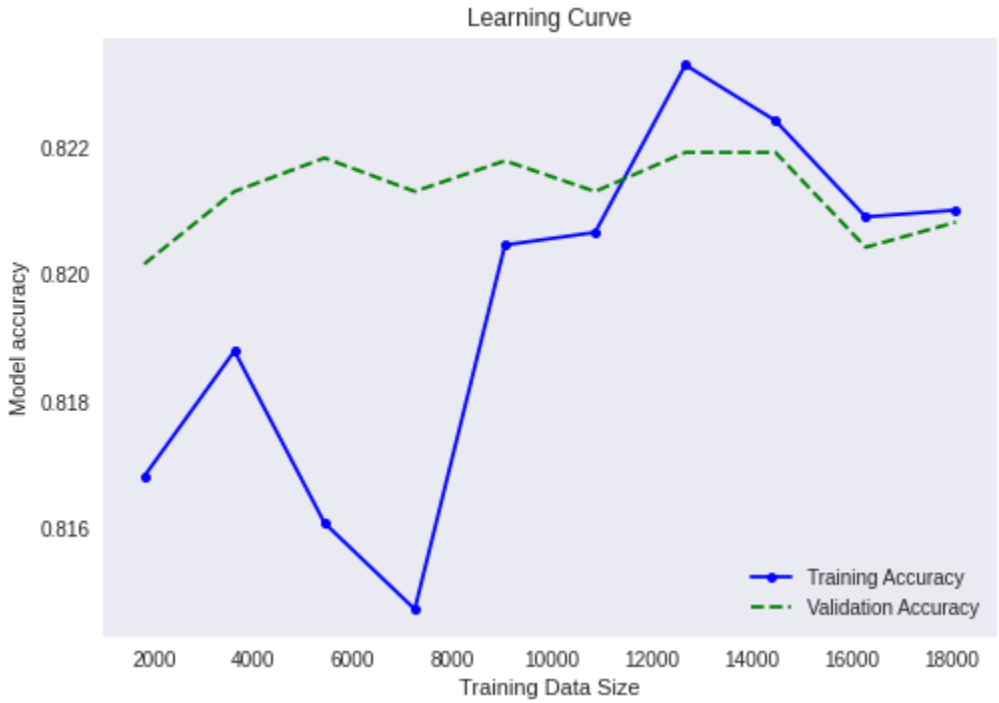


-
- Looking at the confusion matrix we can infer that the model is able to predict the 'ABOVE50K' examples pretty well but it lags in predicting the 'BELOW50K' class accurately.

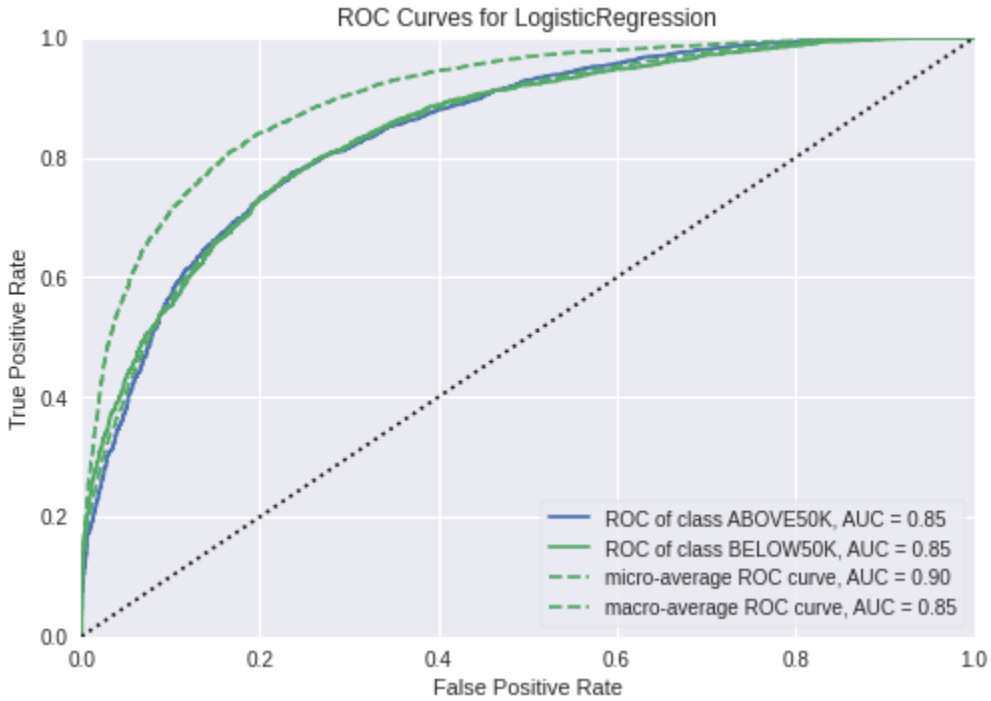
- **Graphs:**

We can see that as the number of training examples increases, the training error increases exponentially and there is a gradual increase in validation error.

The model is a good fit as the validation error decreases and the model converges at around 1000 training size.



The model works well and is a good fit as the accuracy increases as the number of training examples increases

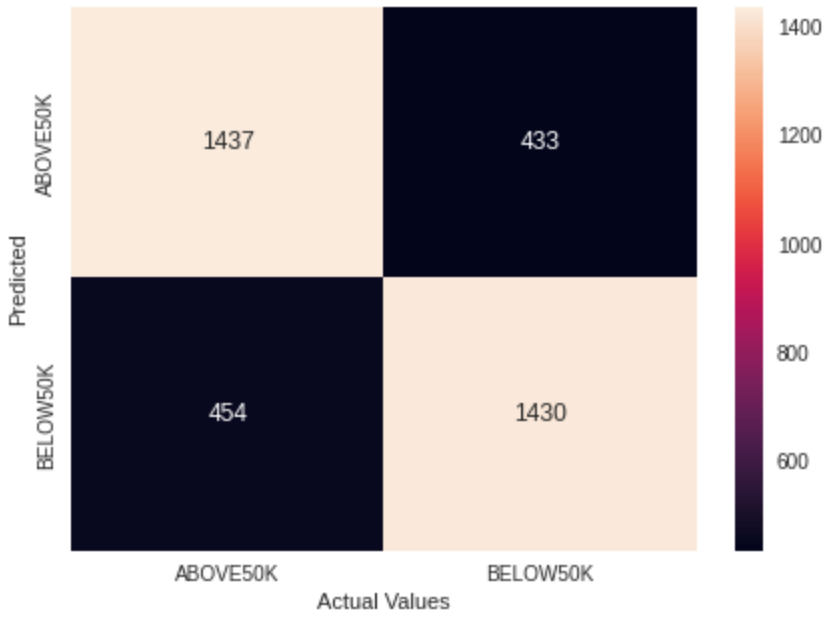


- **Results:**

	precision	recall	f1-score	support
0	0.84	0.94	0.89	5656
1	0.72	0.46	0.56	1885
accuracy			0.82	7541
macro avg	0.78	0.70	0.72	7541
weighted avg	0.81	0.82	0.81	7541

➔ Logistic regression with equal number of training examples of both classes:

- There are 7508 samples of class 1 so we take that amount of samples from class 0 and create another dataset.
- Next, we use the sklearn library to split the dataset into two parts - training data and test data.
- For validation, there is no validation set created as I have used the cross validation method which randomly makes bins of around 20% of training data and then uses these bins to validate the training set.
- Then logistic regression is performed on the data with the following parameters : `logisticregression = LogisticRegression(C=100, max_iter=100,penalty='l2', solver='newton-cg').fit(X_train, y_train)`
  `#c = 1/ lam`
- Confusion matrix:

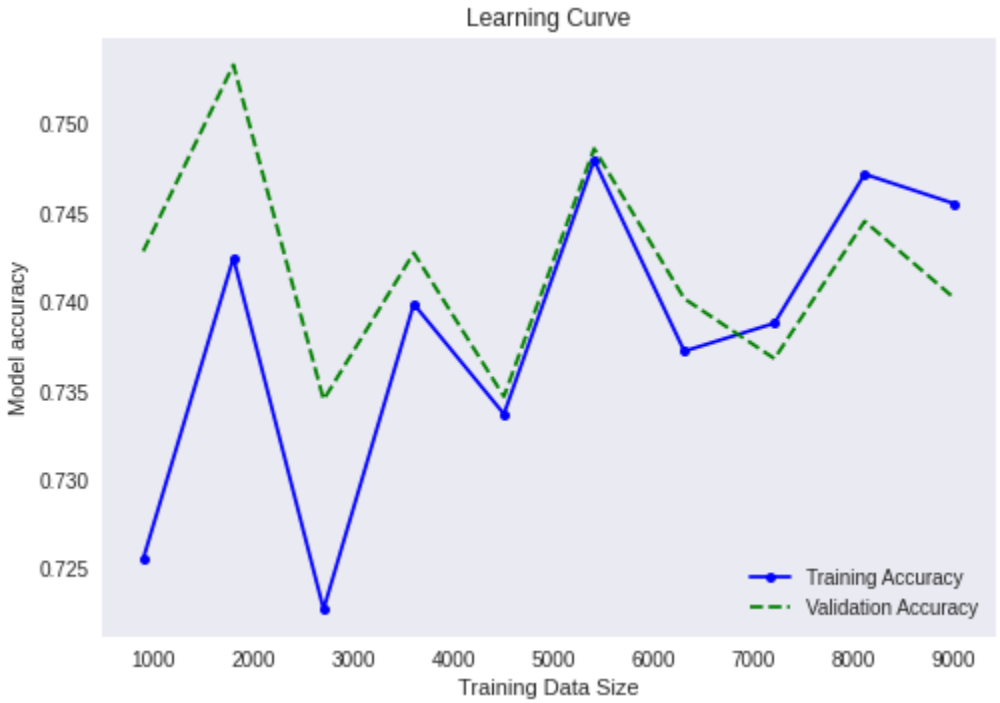


- Looking at the confusion matrix we can infer that the model is able to predict the' examples of both the classes pretty well.

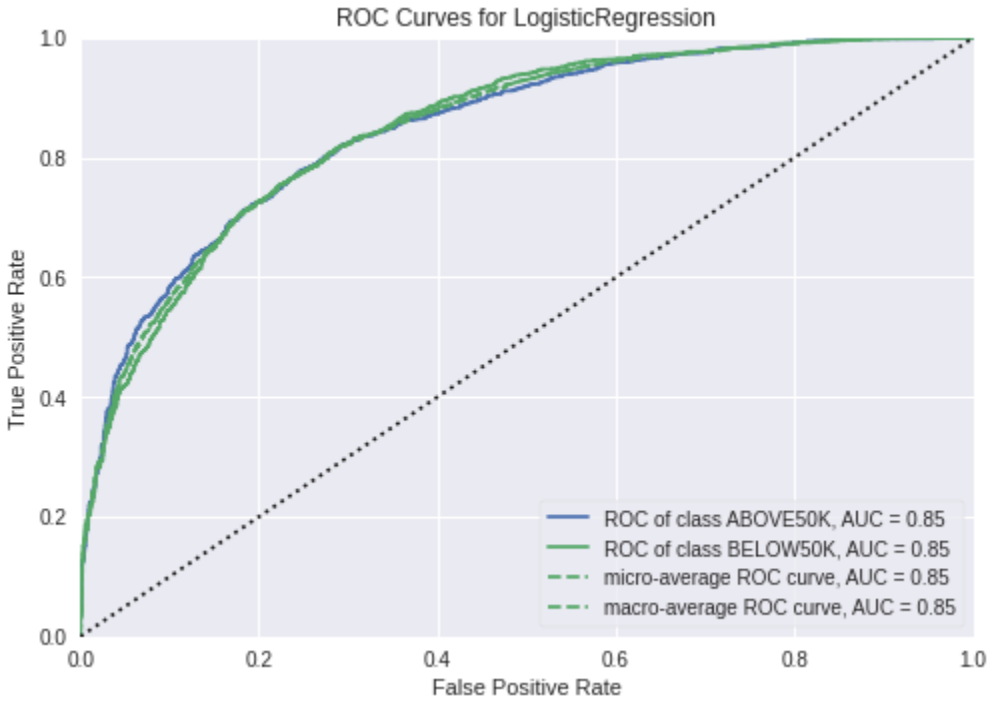
- Graphs:

Upon looking at the above graph of Mean Squared Error vs Training Size we can say that the graph as the number of training examples reach around 8000, the model performance increases and the Mean Squared Error decreases(MSE).

Upon looking at the above graph of Loss vs Training Size we can say that the graph is a good fit as the curves are converging as the training size increases.



Upon looking at the above graph of Accuracy vs Training Size we can say that the graph is a good fit as well as the curves are converging as the training size increases and the accuracy is increasing.

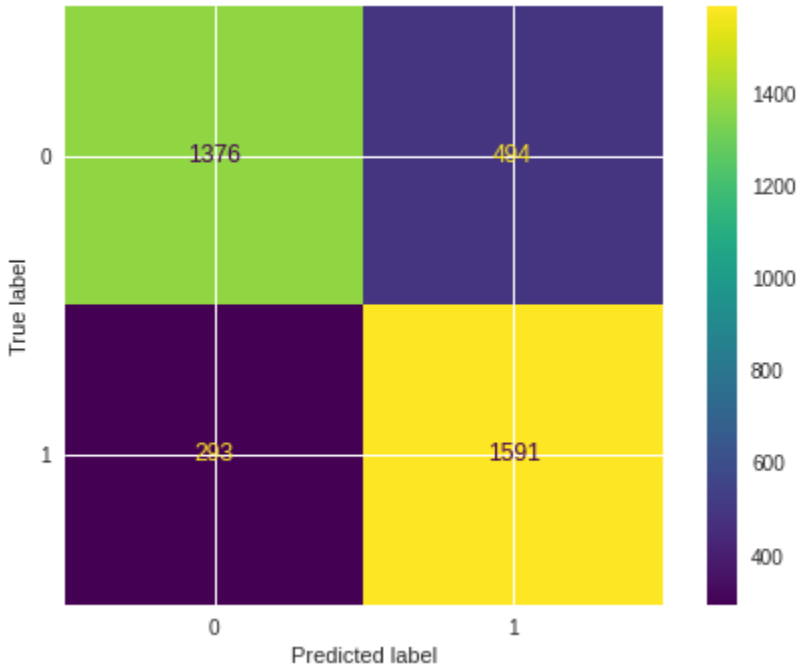


- **Hyperparameter tuning:** Hyperparameters are tuned using the GridSearchCV and RepeatedStratifiedKfold libraries and the best parameters are used to train the model which gave the below outputs.
- **Results:**

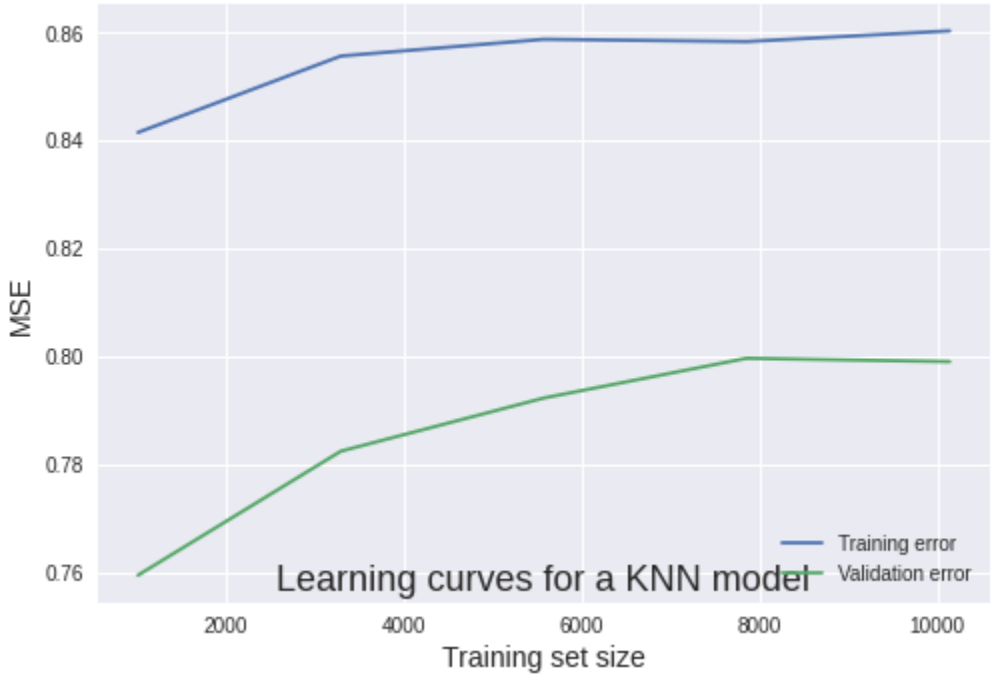
	precision	recall	f1-score	support
0	0.76	0.77	0.76	1870
1	0.77	0.76	0.76	1884
accuracy			0.76	3754
macro avg	0.76	0.76	0.76	3754
weighted avg	0.76	0.76	0.76	3754

❖ Model-2: K-Nearest Neighbors

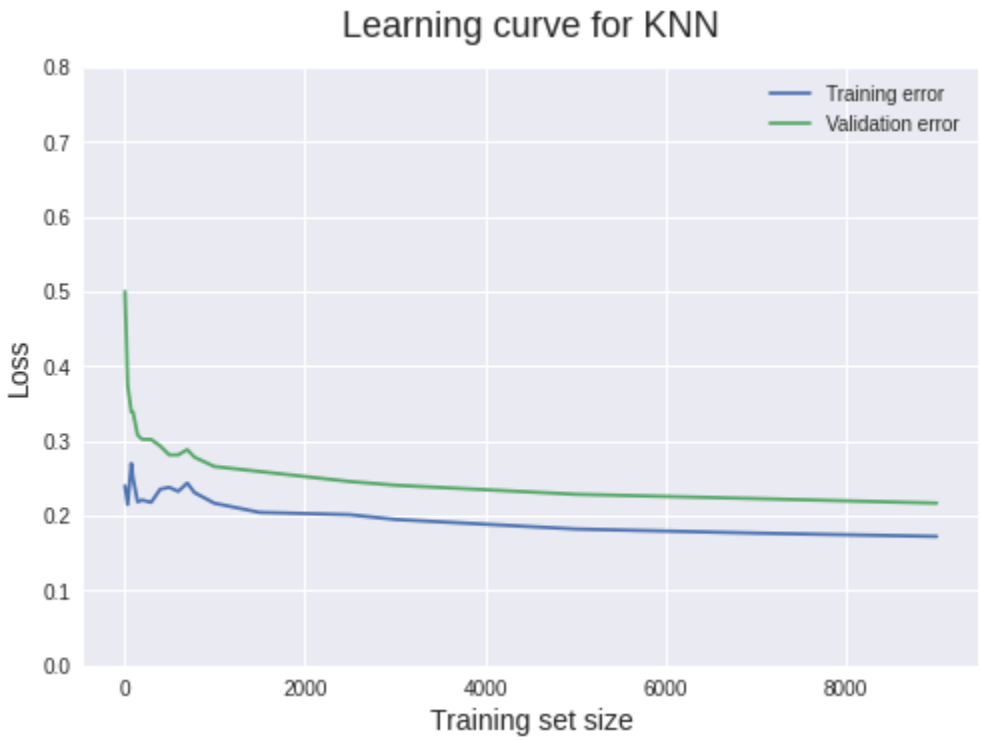
- ❖ Firstly, we find the best value of 'k' which comes to 9.
- ❖ Then we train the KNN model using k=9.
- ❖ **Confusion matrix:**



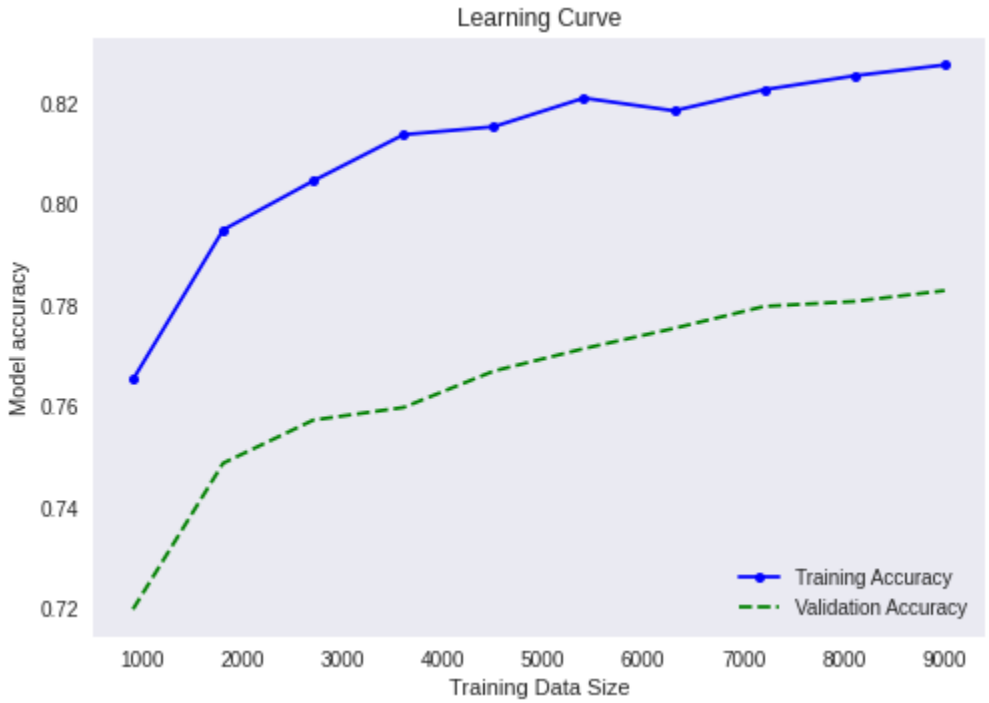
- ❖ **Graphs:**



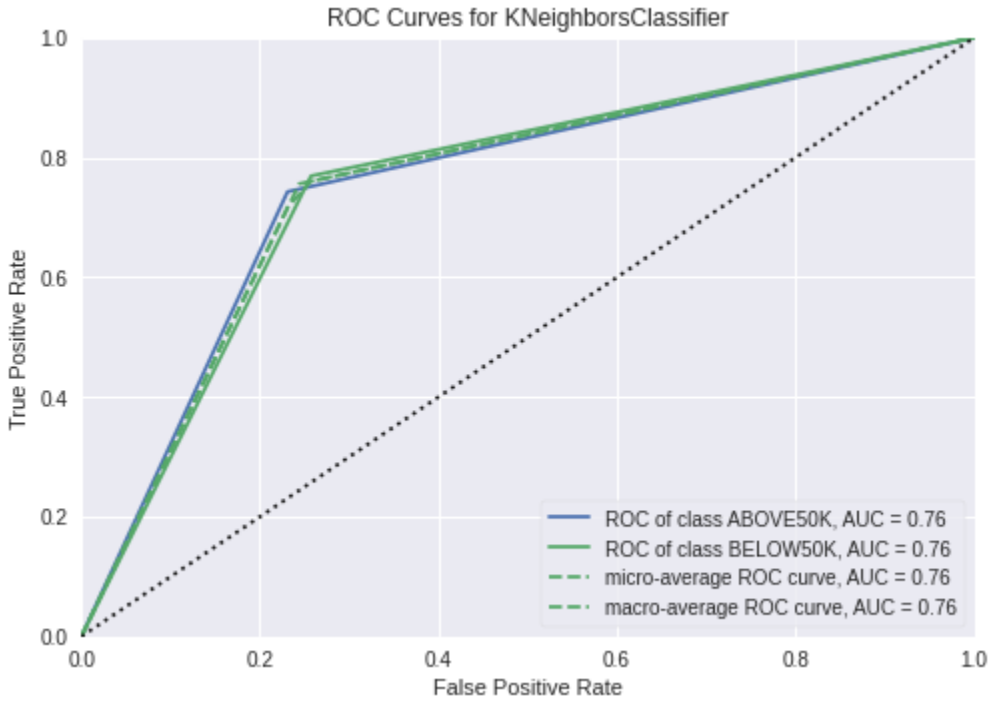
The MSE vs Training set size graph shows that as the number of training examples increases the training and validation errors increase gradually. Looking at the graph we can see that the model is underfit.



The Loss vs Training size graph shows that the loss decreases as the number of training examples increases but the model does not converge.



Looking at the Accuracy vs Training size graph we notice that the accuracy increases as the training examples increase.



❖ **Results:**

	precision	recall	f1-score	support
0	0.76	0.77	0.76	1870
1	0.77	0.76	0.76	1884
accuracy			0.76	3754
macro avg	0.76	0.76	0.76	3754
weighted avg	0.76	0.76	0.76	3754

❖ Conclusion

- ❖ As we can see in the results of all the models, they all give almost the same results for equally sampled data.. So for the given dataset I can infer that both the models - Logistic Regression and KNN work almost identically giving similar results.
- ❖ If the dataset has more training examples of both the classes and the bias is reduced from the original dataset then the models will probably perform better.
- ❖ A multi-layered Neural Network might be able to give a better output as the learnings might increase layer by layer, so the final prediction power would increase.

❖ References

- ❖ Original dataset (UCI Machine Learning Repository) - https://archive.ics.uci.edu/ml/datasets/census+income