```
In [3]: import pandas as pd
```

```
In [4]: from pycaret.classification import setup, compare_models, evaluate_model, s
```

```
In [5]: churn_data = pd.read_csv('/content/updated_churn_data (1).csv')
```

```
In [6]: clf = setup(data=churn_data, target='Churn')
```
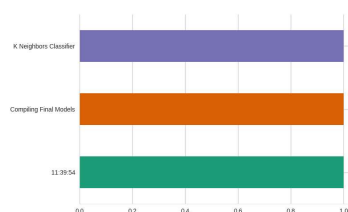
	Description	Value
0	Session id	1150
1	Target	Churn
2	Target type	Binary
3	Original data shape	(7043, 8)
4	Transformed data shape	(7043, 13)
5	Transformed train set shape	(4930, 13)
6	Transformed test set shape	(2113, 13)
7	Numeric features	4
8	Categorical features	3
9	Preprocess	True
10	Imputation type	simple
11	Numeric imputation	mean
12	Categorical imputation	mode
13	Maximum one-hot encoding	25
14	Encoding method	None
15	Fold Generator	StratifiedKFold
16	Fold Number	10
17	CPU Jobs	-1
18	Use GPU	False
19	Log Experiment	False
20	Experiment Name	clf-default-name
21	USI	bfd1

```
In [7]: best_model = compare_models()
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
knn	K Neighbors Classifier	0.7673	0.7055	0.4013	0.5904	0.4769	0.3346	0.3453	0.1170
lr	Logistic Regression	0.7347	0.8224	0.0000	0.0000	0.0000	0.0000	0.0000	0.6670
dt	Decision Tree Classifier	0.7347	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0920
ridge	Ridge Classifier	0.7347	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0870
rf	Random Forest Classifier	0.7347	0.6271	0.0000	0.0000	0.0000	0.0000	0.0000	0.2920
qda	Quadratic Discriminant Analysis	0.7347	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0870
ada	Ada Boost Classifier	0.7347	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0900
gbc	Gradient Boosting Classifier	0.7347	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.3090
lda	Linear Discriminant Analysis	0.7347	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0910
et	Extra Trees Classifier	0.7347	0.6268	0.0000	0.0000	0.0000	0.0000	0.0000	0.2320
xgboost	Extreme Gradient Boosting	0.7347	0.7225	0.0000	0.0000	0.0000	0.0000	0.0000	0.1200
lightgbm	Light Gradient Boosting Machine	0.7347	0.5176	0.0000	0.0000	0.0000	0.0000	0.0000	0.5690
dummy	Dummy Classifier	0.7347	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0860
svm	SVM - Linear Kernel	0.7197	0.0000	0.3300	0.5075	0.3342	0.2007	0.2401	0.1080
nb	Naive Bayes	0.6811	0.8157	0.8617	0.4478	0.5891	0.3687	0.4221	0.1390

```
Processing:    0%|            | 0/65 [00:00<?, ?it/s]
```

Categorical distributions


```
In [8]: evaluate_model(best_model)
```

```
interactive(children=(ToggleButtons(description='Plot Type:', icons=('',),
options=(('Pipeline Plot', 'pipelin…
```

```
In [ ]: save_model(best_model, 'best_churn_model')
```

```
In [9]: def predict_churn_probability_whole_dataset():
            model = load_model('best_churn_model')

            predictions = predict_model(model, data=churn_data)

            return predictions
```

```
In [12]: predictions = predict_churn_probability_whole_dataset()
```

Transformation Pipeline and Model Successfully Loaded

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	K Neighbors Classifier	0.8217	0.8522	0.5361	0.7203	0.6147	0.5019	0.5112

```
In [13]: print(predictions)
         customerID  tenure  PhoneService        Contract  \
0        7590-VHVEG       1             0   Month-to-month
1        5575-GNVDE      34             1        One year
2        3668-QPYBK       2             1   Month-to-month
3        7795-CFOCW      45             0        One year
4        9237-HQITU       2             1   Month-to-month
...             ...     ...           ...             ...
7038     6840-RESVB      24             1        One year
7039     2234-XADUH      72             1        One year
7040     4801-JZAZL      11             0   Month-to-month
7041     8361-LTMKD       4             1   Month-to-month
7042     3186-AJIEK      66             1        Two year

                   PaymentMethod  MonthlyCharges  TotalCharges  Churn  \
0               Electronic check       49.650002     49.650002      0
1                   Mailed check       55.573528   1889.500000      0
2                   Mailed check       54.075001    108.150002      1
3      Bank transfer (automatic)       40.905556   1840.750000      0
4               Electronic check       75.824997    151.649994      1
...                          ...             ...           ...    ...
7038                Mailed check       82.937500   1990.500000      0
7039      Credit card (automatic)      96.878471   6975.250000      0
7040            Electronic check       31.495455    346.450012      0
7041                Mailed check       76.650002    306.600006      1
7042   Bank transfer (automatic)      103.704544   6844.500000      0

         prediction_label  prediction_score
0                       0               0.8
1                       0               1.0
2                       0               0.6
3                       0               1.0
4                       1               0.6
...                    ...               ...
7038                    0               1.0
7039                    0               1.0
7040                    0               1.0
7041                    1               0.6
7042                    0               0.8

[7043 rows x 10 columns]
```

Summary

In this week's task, we utilized the PyCaret library to develop a predictive model for customer churn based on a dataset containing various customer attributes such as tenure, phone service, contract details, and payment method. After preprocessing the data and setting up the PyCaret environment, we compared several machine learning algorithms to identify the best-performing model. Upon selecting the optimal model, we evaluated its performance metrics, saved it for future use, and created a function to predict churn probabilities for new data. This streamlined approach facilitated efficient model development and evaluation, enabling businesses to make data-driven decisions to mitigate customer churn and improve customer retention strategies.