

# ML Code - Basics

Thursday, 14 November 2024

12:58 PM

1. Install python first (check version using `python3 --version`)
2. `python3 -m ensurepip --upgrade`
3. `pip3 --version`
4. `pip3 install scikit-learn numpy`

Open intelliJ

- Install python community edition (File -> Settings -> plugins -> python community)
- Create a workspace on selecting the installed python and a project
- Inside bin of newly created
  - Paste any runnable code by creating a file with extension as .py
  - If facing issue with scikit then do the 4th step again inside the project location

Download tensorflow:

- `pip3 install pydotplus`
- `Pip3 install tensorflow==2.18` # as 3 versions are not supported by tf-keras
- `Pip3 install jupyter`
- `Pip3 install tf-keras`
- `Pip3 install --upgrade transformers`
- Go to `cd /Users/sugnanasekaran/MLCourse`
- Type "jupyter notebook" and enter

Verify installed version:

```
pip list | grep -E "tensorflow|keras|transformers"
```

o/p:

```
keras                3.8.0
sentence-transformers 3.3.0
tensorflow            2.18.0
tf_keras              2.18.0
transformers          4.46.2
```

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from xgboost import XGBClassifier
from lightgbm import LGBMClassifier
from sklearn.metrics import accuracy_score
```

Structure:

- Store train and test data
- `y = train_data["Survived"]`
- `features = ["Pclass", "Sex", "SibSp", "Parch"]`
- `X = pd.get_dummies(train_data[features])`
- `X_test = pd.get_dummies(test_data[features])`
- `models = { "RandomForest": RandomForestClassifier(n_estimators=100, max_depth=5, random_state=1), "DecisionTree": DecisionTreeClassifier(max_depth=5, random_state=1), "XGBoost": XGBClassifier(n_estimators=100, max_depth=5, learning_rate=0.1, random_state=1), "LightGBM": LGBMClassifier(n_estimators=100, max_depth=5, learning_rate=0.1, random_state=1) }`
- `for name, model in models.items():`
  - `model.fit(X_train, y_train)`
  - `predictions = model.predict(X_test)`
  - `print(f'{name} Accuracy: {accuracy_score(y_test, predictions)}')`

Keggal output:

```
output = pd.DataFrame({'PassengerId': test_data.PassengerId, 'Survived': predictions}) output.to_csv('submission.csv', index=False)
```

```
index=raise)
print("Your submission was successfully saved!")
```

Input read:

```
import pandas as pd
import numpy as np
def read_csv(path):
    return pd.read_csv("/kaggle/input/store-sales-time-series-forecasting/" + path)
Train_data = read_csv("train.csv")
Test_data = read_csv("test.csv")
train_data = pd.read_csv("/kaggle/input/titanic/train.csv")
train_data.head()
test_data = pd.read_csv("/kaggle/input/titanic/test.csv")
test_data.head()
```

Output submission:

```
end_data = data_test[['id']].copy() # Copy the 'id' column as a DataFrame
end_data['sales'] = predict_y_xg # Add the predicted sales as a new column
Or
sample_submission = pd.read_csv("/kaggle/input/nlp-getting-started/sample_submission.csv")
sample_submission["target"] = clf.predict(test_vectors)
sample_submission.head()
sample_submission.to_csv("submission.csv", index=False)
```

# Save to a CSV file

```
end_data.to_csv('/kaggle/working/submission.csv', index=False)
```

Example of multiple models and finding best one -> <https://www.kaggle.com/code/mahmouddevgamal/getting-started-with-titanic>